TELERADIOLOGY KIT WITH AUTOMATIC BRAIN SEGMENTATION

by

SANDHITSU RANJAN DAS

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY KANPUR

April, 1999

TELERADIOLOGY KIT WITH AUTOMATIC BRAIN SEGMENTATION

A Thesis Submitted

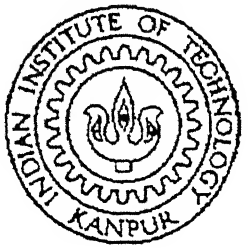
in Partial Fulfilment of the Requirements

for the Degree of

Master of Technology

by

SANDHITSU RANJAN DAS

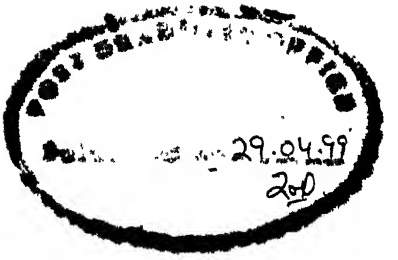


to the

DEPARTMENT OF ELECTRICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY, KANPUR

April, 1999



Certificate

It is certified that the work contained in the thesis entitled "TELERADIOLOGY KIT WITH AUTOMATIC BRAIN SEGMENTATION", by Sandhitsu Ranjan Das, has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

(Dr. G. C. Ray)
Professor
Department of Electrical Engineering
Indian Institute of Technology, Kanpur

April, 1999.

Acknowledgements

I am extremely grateful to Dr. G.C. Ray, who has helped shape my academic career over the years. It is he who introduced me to the world of Biomedical Engineering. I have immensely enjoyed my maiden research project with him.

I am thankful to Dr. Stella Atkins of Simon Fraser University, Canada for providing me with MRI data sets.

My two years of M.Tech. have been - in more ways than one - complementary to my undergraduate years. And the person responsible for this was Antu. It was great to be with him. Alongwith Siddhartha, he provided intellectual company to me. I learned a lot from them, and a lot remain to be learned. I respect them.

The E-midders group - Antu, Ritesh, Karthik, Ashokji, Jay and Pratul were the ones who made these two years such an enjoyable experience. I love them.

A great deal of my life in the last two years revolved around Linux, and *sadhana*, our Linux server. Antu and I played with it, learned a lot and maintained it with a great deal of zeal. And if there is one person who has been the greatest inspiration behind all these, it was Shafi, our *Speak O' Guru*.

Eating out in the company of IEEE (Institute of Excessive Eating Engineers) was a frequent heavenly gastronomical experience. I would like to thank my fellow members Antu, Siddhartha, Mama, Bnaru, Deepanjan, Kousik and Mashuq.

Abstract

The tremendous growth of information technology in the recent years has made it possible to design a variety of telemedicine systems. Several large scale multinational projects are currently being pursued which is revolutionizing remote health care. Teleradiology is one area where many projects are being undertaken. In India, there have been very few such efforts mainly because of lack of proper IT infrastructure and prohibitive costs of large scale telemedicine systems.

In this work, we develop an inexpensive teleradiology kit which can run over any TCP network. It includes a client program providing a graphical user interface to the doctor who can retrieve data from a server site where MRI brain slices of patients are available. Other features implemented include remote authentication, automatic brain segmentation by the server, knowledge-based image compression prior to data transmission, support for multiple patients and data caching.

Brain segmentation is a prerequisite of many important MRI post-processing algorithms. It also allows for data compression. The automatic brain segmentation algorithm implemented on the server site is based on existing automatic thresholding and morphological methods. The algorithm has been modified to improve computational efficiency.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation for Remote Radiologist

Information technology is a discipline that defies all bounds and barriers and has already demonstrated the most amazing developmental "curve" that no other discipline has ever witnessed or generated during what sounds like a mere "nano" time span. While the internet has captured much of the world's attention and continues to do so, the issues surrounding its functionalities, as well as its inevitability as an "everyday" tool in the future, are becoming more complex. Network traffic congestion, information security, address space and high bandwidth requirements are just a few examples. The continued search for newer solutions for addressing these issues has resulted in the evolution of Internet's other "adjuncts". In this framework, one talks about projects and protocols such as the Internet II, the very-high-speed backbone network services (vBNS), the next generation Internet Protocol IPv6 etc.

Two major disciplines that have been and will continue to benefit from such developments are medicine and health care. Numerous biomedical applications have already expounded on the role and impact of the "digital future" that medicine and health care hold [1]. With the emergence of a wide class of enabling technologies such as fiber optic systems, integrated services digital networks (ISDN), asynchronous transfer mode (ATM) etc., we have already seen the proliferation of these technologies in a number of health care and biomedical engineering projects. Examples include telemedicine, teleradiology, hospital-physician networks, health care delivery systems and application of virtual reality and telerobotic technologies.

1.1.1 An International Perspective

Computing and communications technologies have advanced to the point where the world is gradually moving toward collaborative medicine and collaborative research through distributed system capabilities over high-speed networks and new multimedia and video conferencing technologies. The various National Information Infrastructure and Global Information Infrastructure (NII/GII) concepts that are constantly evolving in several countries, along with major telemedicine projects jointly being pursued by several countries are just pointers to this fact.

Examples of such multinational efforts are:

- Several projects under the auspices of European Commission (EC) like the TeleSCAN program which demonstrates the benefits of multimedia telematics to the European oncologist.
- SuperJANET: This is the high bandwidth Joint Academic Network in the

UK which is exploited in several major projects like neurosurgery planning, Interactive Teaching program in surgery etc.

- Medical and health care services based on the World Wide Web (WWW). Examples include Visible Human Data (VHD) [2] which is a huge digital database of MRI and CT images and Visible Embryo Project [3] whose aim is to digitize sets of serial microscopic cross sections through human embryos.

- The new generation teleradiology system called CHILI [4] being developed in Germany. At the core of the system is a general radiological workstation that possesses additional functions for teleradiology.

- Telemedicine remote consultation and diagnosis (RCD) software developed in the University of Arizona [5]. RCD allows physicians to collaborate on radiology and pathology cases from distributed geographical locations. They have used object-oriented design methodology to design and develop a software system in modular fashion.

1.1.2 The Indian Scenario

In India, however, the effect of such a sweeping change in the field of health care is not as visible. The concept of NII has been adopted by the National Informatics Center's Network (NICNET) which presently connects 15 major cities and about 500 districts. But mainly because of the prohibitive costs of expensive technologies such as high-speed dedicated ISDN lines, there have been few or almost no major project initiatives.

However, if we carefully look into the huge amount of information as well as

tools still available to us at a relatively lower cost through the internet, we can build inexpensive health care systems, which can actually improve the quality of health care as well as reduce the overall costs of the system. This is what has been the motivation behind this work.

We have designed a simple client-server application where the client software is a graphical user interface (GUI) at the doctor's site whereas the server runs at a remote patient site where MRI scans of patients are taken. Not only can the doctor (the human radiologist) view the scans through the GUI, he has the option of applying several automated algorithms to it. Whereas these post-processing algorithms have not been implemented as part of the kit, automatic brain segmentation at the server site has been implemented using a fast algorithm adapted from [6].

1.2 Motivation for Brain Segmentation

Magnetic resonance imaging (MRI) is a noninvasive method for producing three-dimensional (3D) tomographic images of the human body. MRI is most often used for the detection of tumors, lesions, and other abnormalities in soft tissues, such as the brain. Clinically, radiologists qualitatively analyze films produced by MRI scanners.

Recently, computer-aided techniques for analyzing and visualizing magnetic resonance (MR) images have been investigated. Many researchers have focused on detecting and quantifying abnormalities in the brain. Automatically identifying the brain in MR images of the head is an important step in this process. Most often the radiologist is interested only the portion of the image which encloses

the brain.

1.2.1 Multiple Sclerosis Lesion Segmentation

Multiple sclerosis (MS) is an autoimmune disease characterized by damage of the myelin covering of neurons in cerebral white matter. The damaged areas or lesions are distinctly visible in MR images. For this reason, MRI is used to detect and track MS lesions in the brain. As MS progresses, the number and size of lesions in the sufferer's brain increases. During treatment and clinical studies, doctors use MRI scans to monitor this change in lesion volume. The monitoring process involves painstakingly outlining every lesion in scans of possibly hundreds of patients.

Therefore, researchers are investigating methods for automatically segmenting MS lesions in MR scans of the head. One such popular method is Johnston's segmentation method [7] which uses a stochastic, Bayesian relaxation based algorithm called iterative conditional modes (ICM). Using tissue intensity histograms from manually identified regions in the MR scan, it attempts to isolate all tissues in the brain, including MS lesions, white matter, grey matter, and cerebral spinal fluid (CSF). Theoretically, it can also isolate tissues outside the brain, such as the eyes, skin, fat, and the skull.

Because the distinct tissue types are represented by non-unique intensities in the MR scans, the ICM algorithm confuses tissues outside the brain with tissues inside the brain. For example, the intensity of the eyes could be identical to the intensity of some MS lesions. Further, the speed of the algorithm is proportional to the number of tissues it segments. It is therefore desirable to isolate

the brain in MR images before applying the ICM algorithm. This is another motivation behind the pre-processing of MR images in terms of automatic brain segmentation.

1.2.2 Image Compression

A single clinical MRI scan occupies several megabytes of disk space. Effective image compression schemes are important for storing multitudes of scans. With the advent of teleradiology, where MRI scans are transmitted by wire to remote sites for evaluation by specialists, MR image compression plays a huge role in increasing transmission speed [8, 9]. Moreover, in MRI scans of the head, doctors are usually more interested in the brain as opposed to the region outside the brain. For this reason, one can develop a lossy MRI compression scheme that selectively compresses the region outside the brain at a higher compression ratio than the brain. Thus, one can achieve high compression ratios while maintaining image quality in the brain area. Obviously, automatic intracranial boundary detection is a prerequisite for such a scheme.

This is the case with our teleradiology kit also. Since brain segmentation allows one to reduce the amount of data in the images slices, it results in significant savings in bandwidth, and can allow for the design of inexpensive telemedicine systems in our country.

This thesis implements an automatic method for intracranial boundary detection in MR images of the head at the server site. The intracranial boundary is the boundary between the brain and the intracranial cavity. It accurately segments the brain from other features in the head.

1.2.3 Available algorithms of brain segmentation

The survey by Clarke *et al.* of segmentation methods for MR images [10] describes many useful image processing techniques. Brain segmentation schemes can be divided into several groups: threshold based methods, refinement of brain contours, statistical methods and region growing methods.

Brain extraction using automatic thresholding

Suzuki and Toriwaki use iterative thresholding to distinguish brain brain tissues from others in axial MR slices [11]. This method is ineffective in the presence of Radio Frequency (RF) inhomogeneity.

Brummer *et al.* use histogram analysis and morphology to generate a 3-D brain mask [12]. Using a model of background noise, they first automatically generate a mask of the head. Then they create a brain mask using an automatic threshold based on a presupposed brain voxel intensity distribution. Mackiewich *et al.* [6] also adapted this technique. In our kit, we have modified this algorithm to suit the needs of a remote telemedicine application, so that the algorithm is fast and allows for data compression.

Refinement of brain contour

Chakraborty *et al.* combine statistical segmentation and boundary detection to isolate features in MR images [13]. They first segment the images using a method similar to the iterated conditional modes algorithm [7]. They then use a parametrically deformable shape model algorithm to find the boundary of interesting

features in the segmented image.

Lundervold and Storvik proposed a segmentation for brain parenchyma in the central slices of multispectral MR images [14] which uses a model-based segmentation method and also uses a new Bayesian dynamic contour (BDC) model to detect the boundaries.

Statistical Methods for brain segmentation

Kapur *et al.* [15] combine Well's statistical classification of MR images [16] with image processing methods, to segment the brain in 3-D gradient-echo MR images.

Cline *et al.* segment the brain from MR images of the head using statistical classification [17]. To segment the brain, samples of brain voxels and non-brain voxels are interactively i identified. Bivariate normal distributions, corresponding to the different tissue types in the PD-weighted and T2-weighted MR images, are fitted to the samples intensities. All the image voxels are then classified according to their intensities lie in the distributions. This method requires user interaction, and falsely classifies non-brain regions such as eyes, as brain.

Region Growing and Boundary Detection

Pannizzo *et al.* detect the intracranial boundary in axial MRI slices by tracing a horizontal line outwards from the center of the image [18]. The point, in each direction, at which the voxel intensity under the line drops below a reference threshold is considered to be a point in the intracranial boundary. A running average of voxel intensities under the brain is then computed. The intracranial

boundary points are relocated to the first voxels with an intensity too far below the average. The running-average procedure is repeated for rows in the image. The entire process is then repeated for all columns. The result is a sequence of points defining the intracranial contour.

Chapter 2

Remote Radiologist Interface

2.1 Overview

A simple Graphical User Interface (GUI) based client-server application has been developed in this thesis for the purpose of teleradiology. The *client site* is also the *Doctor site* where the GUI application is run. The *server site* is the *patient site* where the MRI scan acquisition takes place. The doctor connects to the patient site through the GUI application and requests for MRI slices. After intracranial boundary detection at the client end, the images are sent to the doctor (server) end, where the doctor can apply several post-processing algorithms (not implemented in this thesis).

2.2 Client-Server Model of Networking

The late 1980's had seen a major paradigm shift in the way people use computers. A new model of computing was born, known as *client-server computing*. In a client-server environment, one or more powerful, centralized machines, known as *servers*, provide a variety of services to individual workstations, or clients. For example, *file servers* provide common storage for user files, and users access them through a number of different protocols. An *anonymous ftp server* in the internet typically stores lots of useful software packages to be downloaded by people around the world through several client softwares like *ftp* or *netscape*. Obviously, this model of computing became more and more popular with the proliferation of the internet. Application softwares in any conceivable field were being written as client-server applications, which not only ensured optimum usage of computing resources in a networked environment, but also allowed users to get access to information residing on a remote machine through a well-defined programming interface. The telemedicine applications are inherently network-based applications, and are therefore always based on client-server computing.

2.3 Inter-Process Communication between Client and Server

The communication between the client and the server is actually the communication between two (or more) processes (programs) running on different machines. Therefore, the *server* is actually synonymous to the *server process* running on the server machine, and the *client* is synonymous to the *client process* running on the client machine.

Now, the server and the client programs must communicate with each other. In our case, the GUI-based application running at the doctor's site must communicate with some server program at the patient site.
How does this happen ?

There are several kinds of programming facilities available for achieving the above, together known as the Inter-Process Communication (IPC) facilities [19, 20]. The basic idea of this programming interface is to make IPC similar to file Input/Output (I/O). In a UNIX system, a process has a set of I/O descriptors, from which one reads and to which one writes.[1] Descriptors may refer to normal files, to devices (including terminals), or to *communication channels*. This identification of IPC with I/O is quite longstanding in the UNIX system and has proved quite successful. At first, however, IPC was limited to processes communicating within a single machine. With 4.2 BSD (Berkeley Software Distribution) version of the UNIX operating system, however, this has been extended to include IPC between machines.

The most common IPC facilities available in the UNIX system are signals, pipes, shared memory, semaphores, message queues and socket-based IPC [20]. We have used socket-based IPC to implement the communication between the patient and doctor ends.

[1]Note: We shall limit our discussion to UNIX systems only since this is the operating system of choice in a networked environment. However, some of the IPC facilities are now also available in Microsoft-based operating systems.

2.4 Socket-Based IPC

Sockets [21] provide a programming interface which maybe used for network communications. A *socket* is a communication endpoint and represents an abstract object that a process may use to send or receive messages. With socket-based IPC. one has a choice of mode of communication. Communication is either through a *stream* socket or by *datagram.* Stream communication implies a connection. The communication is reliable, error-free, and no message boundaries are kept. Reading from a stream may result in reading the data sent from one or several calls to `write()` or only part of the data from a single call, if their is not enough room for the entire message, or if not all the data from a large message has been transferred. The protocol implementing such a style will retransmit messages received with errors. Datagram communication does not use connections. Each message is addressed individually. If the address is correct, it will generally be received, although this is not guaranteed.

A protocol is a set of rules, data formats, and conventions that regulate the transfer of data between participants in the communication. In general, there is one protocol for each socket type (stream, datagram etc.). Since our application must transfer data reliably, we have used a streamed socket type. And we have used the Transmission Control Protocol (TCP) which is the default protocol for connection-oriented sockets.

2.5 General System Architecture

In a connection oriented protocol, the server program first opens a communication endpoint (a socket) by calling `socket()`, then binds it to a port[2] through `bind()`. It then calls `listen()`, in which it blocks until a client requests a connection. Meanwhile, a client program, usually on a different machine, calls `socket()` and `bind()`, followed by a `connect()` to connect to the server. The client blocks in `connect()` until the connection is established.

When the connection request arrives at the server machine, `listen()` returns, and the server calls `accept()` to accept the connection. This sends a reply to the client, which returns from `connect()`. The connection is now established. The server, in general, now sits in a loop, calling `rcv()` or `read()` to receive a client request, processing it, and calling `snd()` or `write()` to send a reply. Similarly, the client calls `snd()` or `write()` to send a request and `rcv()` or `read()` to receive a reply. This is depicted in Figure 2.1.

2.6 Design of the GUI

The application running at the client (doctor) site as part of our teleradiology kit provides a simple GUI for the doctor to operate. The GUI was developed with the following design objectives in mind:

[2]The port number can be thought of as the number of a mailbox, into which the protocol places one's messages. The system will assign an unused port number when an address is bound (say by `bind()`) to a socket

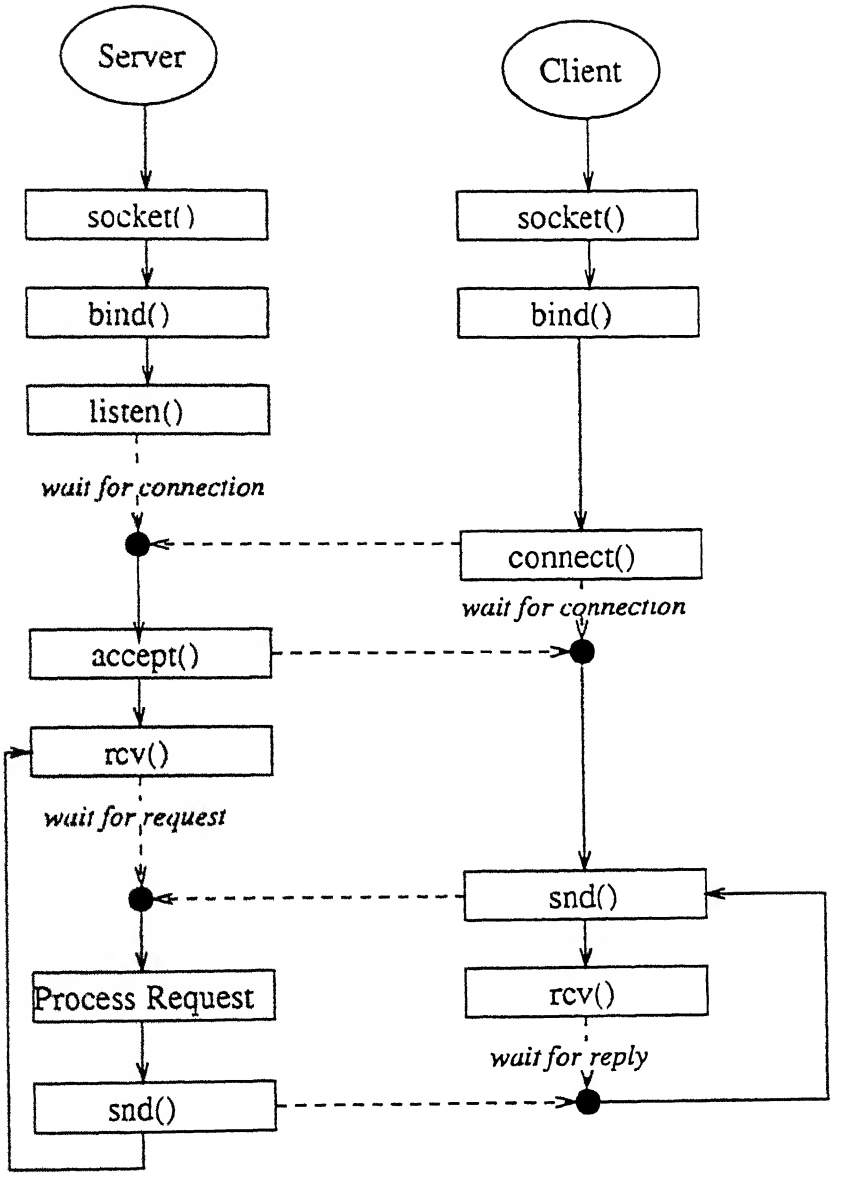


Figure 2.1: System Architecture for TCP socket-based IPC

1. Since it is a networked application, and patient data is confidential, adequate security measures should be implemented for authentication and data transmission.

2. The GUI should provide logical functionalities required of a teleradiology kit, like retrieving data for different patients, the ability to control the preprocessing at the server end etc.

3. Provision for data caching should be provided, since transmission of image slices containing huge amount of data is an expensive operation.

4. The response of the interface should be as fast as possible.

5. The GUI should be aesthetically pleasing.

2.6.1 The choice of Tcl/Tk

Tcl stands for Tool Command Language. It is a general purpose scripting language built as a C library package. Tk stands for Tool Kit, which is an X11 (The X-Window System used in all UNIX systems) toolkit based on Tcl. Tcl and Tk together provide a programming system for developing and using graphical user interface (GUI) applications.

GUI applications in telemedicine intended for use over the network have been generally written in JAVA [5], or C [4]. JAVA seems to be a very popular choice because of its platform independence and extensive libraries available for GUI development. However, JAVA applets running in a browser tend to be extremely slow because of in-built security checking as well as the fact that it is interpreted. Coding in C would always result in a robust and efficient application, but the

downside is the huge effort involved in coding and debugging, particularly for GUI development.

In this respect, Tcl and Tk [22] provide some attractive features catering to our design objectives which prompted us to choose them as our development base.

1. GUI applications can be written entirely as Tcl scripts, using a windowing shell called `wish`. This allows one to program at a much higher level that one would in C, C++ or JAVA. And Tk hides many of the details that C programmers must address.

2. Since Tcl is an interpreted language, it allows rapid application development. One can generate and execute new scripts on the fly without recompiling or restarting the application. Since Tcl is interpreted, it executes more slowly than compiled C code. This is the case with JAVA also. But since Tcl is *embeddable*, the performance-critical parts can always be implemented in C.

3. Tcl makes it easy for applications to have powerful scripting languages. To create a new application, all one needs to do is to implement a few new Tcl commands that provide the basic features of the application. Then one can link one's new commands with the Tcl library to produce a full-function scripting language that includes both the commands provided by Tcl and those that one wrote himself.

Keeping in view the above advantages, we chose Tcl/Tk for the GUI development. As we discuss the GUI in the following sections, this will become clearer. And our teleradiology kit, which is therefore an extension of **Tk**, is called **Bk** (Brain Kit) (Figure 2.2). This is the client software. The server software which

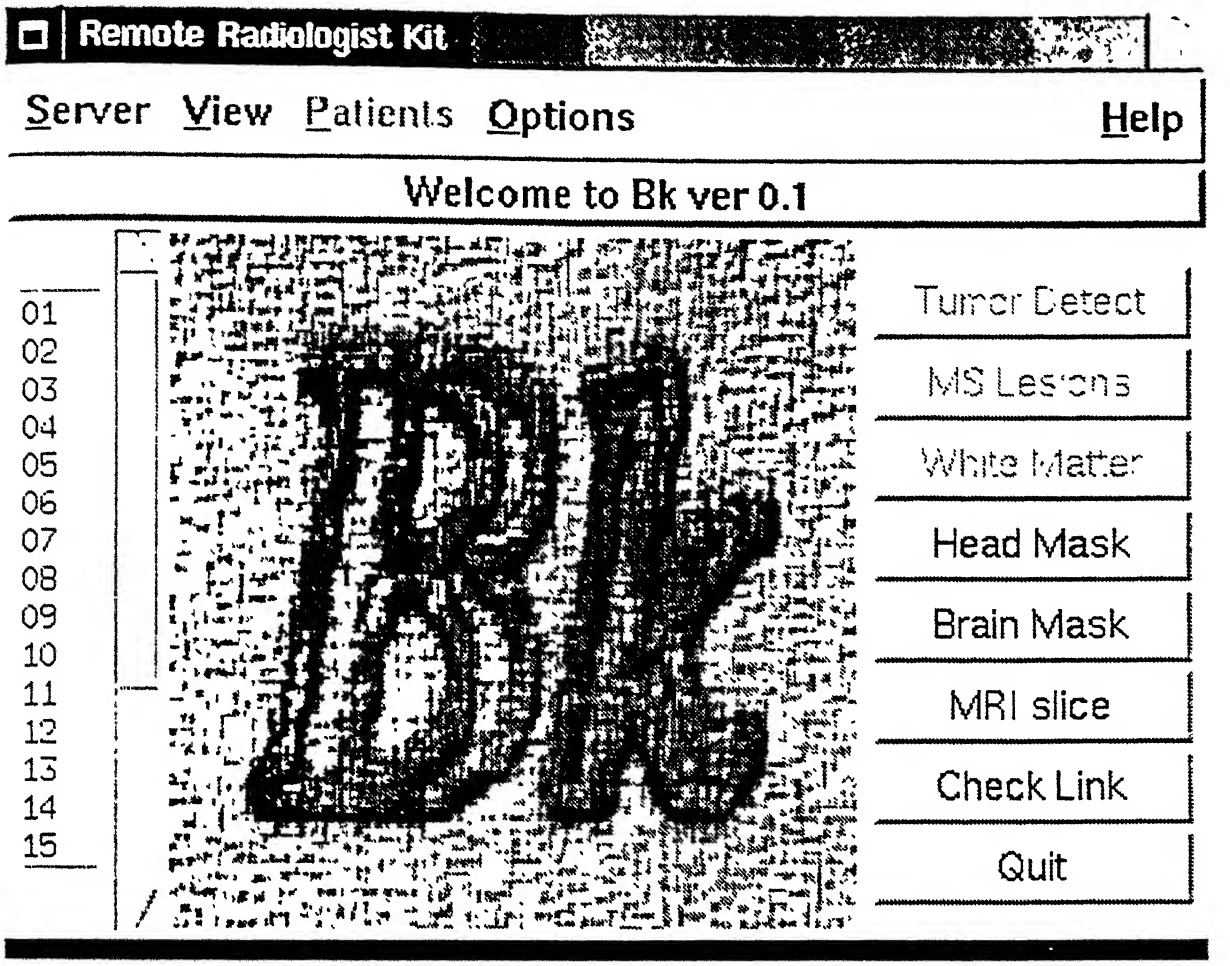


Figure 2.2: The Teleradiology kit Bk. at startup

runs at the patient site is called the *Image Transfer Protocol Server* (ITP server) since it implements a protocol for transferring brain images.

2.7 Working of the Teleradiology Kit, Bk

In this section, we shall discuss the GUI in more detail and look into the various features that have been implemented. **Bk** has been implemented in two layers:

- The graphical user interaction and display of images are handled entirely by Tcl scripts.
- The network programming and image processing is done by code written in C.

This two-layer design optimizes performance, because the computation intensive jobs and critical interaction with the server are always handled by Tcl procedures implemented in C. The C code is compiled as a statically linked shared library and is loaded into the Tcl interpreter at run-time when **Bk** is invoked. This has been possible because of the extensible nature of the Tcl language.

Bk provides the following functionalities to the user (the doctor):

1. It provides a standard UNIX authentication scheme for the user.
2. It supports the handling of MRI scans of multiple patients (upto a maximum of 5 in the current implementation).
3. It allows the doctor to remotely issue commands for head and brain segmentation of the MRI scans.
4. Based on the detected brain boundary or head boundary, it provides a simple image compression technique for the transfer of data slices over the network. This can also be turned off if so desired.

5. It provides dynamic selection of location as well as type (original/head mask/brain mask) of slice to be displayed.

6. It provides optional viewing with detected bounding boxes.

7. It provides a simple data caching mechanism to minimize data transmission by keeping track of already transferred slices so that in case of a repeat request, a slice is not transferred again.

All of the above will be discussed in greater detail in subsequent sections.

2.8 Authentication

Once the GUI application is started by the radiologist at the client machine, the first thing he/she needs to do is to is to make a connection to the ITP server program running on the server machine. For this, Bk needs to know two things: the *name* of the server machine, and the *port no.* at which the server is *listening* for client requests. A default server name is selected by the Tcl script at startup. It can, however, be changed by the `Options` menu item on the menu panel as shown in Figure 2.3.

The port no. of the listening port is coded inside the Tcl script. Instead, it is possible to use the portmapper protocol [19] if the server has registered its service with the portmapper process. But this requires the server to run on privileged ports requiring superuser access. This could trigger a possible security hole in the system, and has been, therefore, avoided in the present implementation.

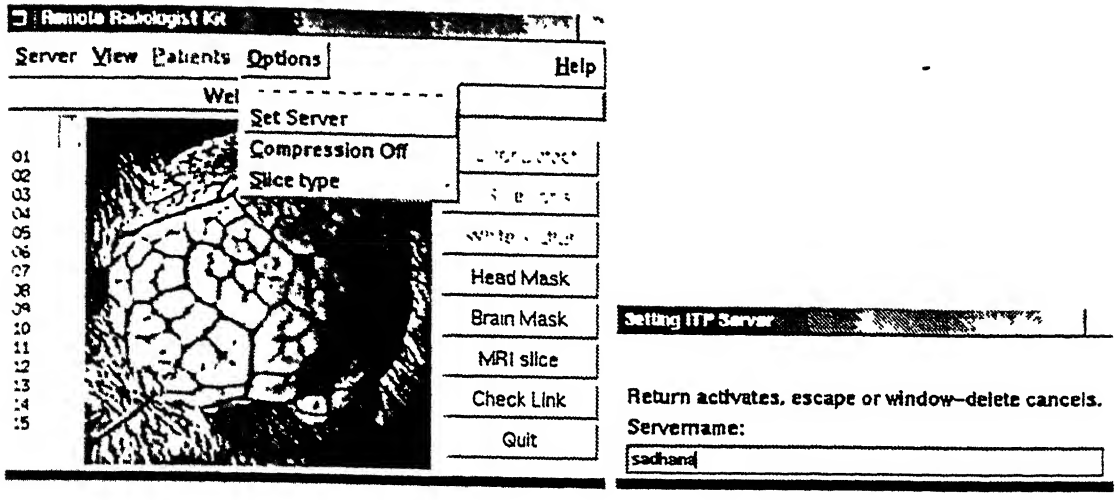
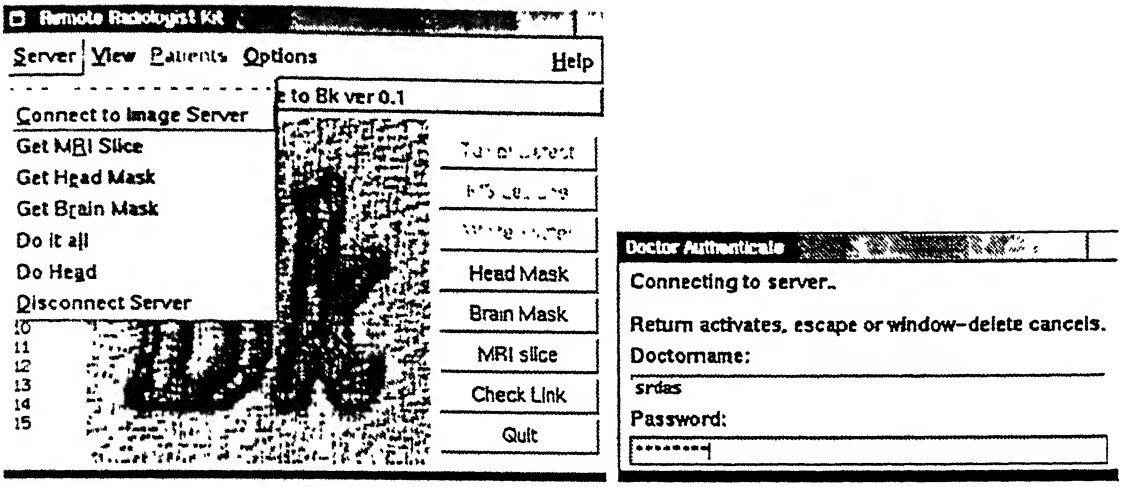


Figure 2.3: The option for changing the server

The connection to the server is made by selecting the `Connect to Image Server` option in the `Server` option menu. The doctor has to fill in his user name and password in the dialog box that pops up (Figure 2.4). The user name and password strings are sent to the server which authenticates the doctor using the Data Encryption Standard Algorithm (DES) [28] used in UNIX systems for password authentication. If the authentication is successful, a connection is established with the server and the client asks for *initialization*, discussed in the next section. Otherwise, the connection is dropped and the socket is closed.

2.9 Initialization Procedure

Once the doctor is authenticated and a connection is established, **Bk** asks for initialization data from the ITP server. This basically consists of the names and total no. of patients whose MRI scans are available for examination. The server



(a) (b)

Figure 2.4: Doctor's Authentication. (a) The Connect option in the Server menu. (b) Authentication window.

finds out the required information and sends it back. Upon receipt of this data, Bk creates the Patient menu, which has so far been deactivated (Figure 2.2). The first patient in the queue is automatically selected for examination (Figure 2.5).

At this point, the doctor is ready to examine the scans of the patient selected.

2.10 Remote Command Buttons

The doctor can issue remote commands through the various buttons available on the right-hand panel of Bk (Figure 2.2). The same commands are also available through the pull down Server menu. These commands are described below:

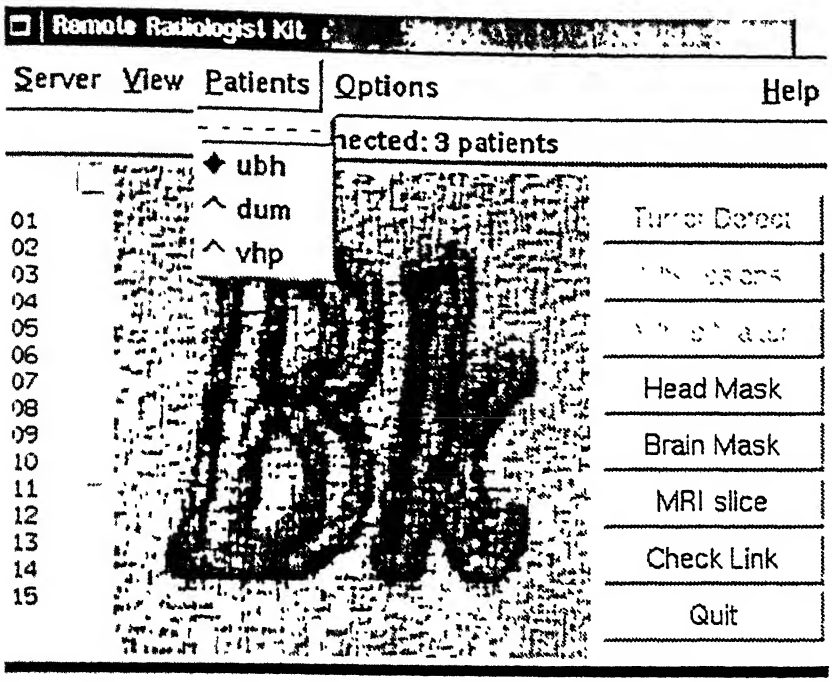


Figure 2.5: The `Patient` option menu activated once the patient list is obtained from the server.

- `Check Link`: This button tests the ITP connection (if already established) to see if the image transfer rates are satisfactory or not. A test image is transferred from the server and displayed (Figure 2.6).

- `MRI Slice`: This button transfers and displays the original MRI slice according to current selections, *i.e.*, the patient selected, the slice type selected (PD,T1 or T2) through the `Options->Slice type` menu item (Figure 2.7) and the slice no. selected from the listbox with scrollbar on the left panel of **Bk**. If compression is turned on, compression according to head mask is performed.

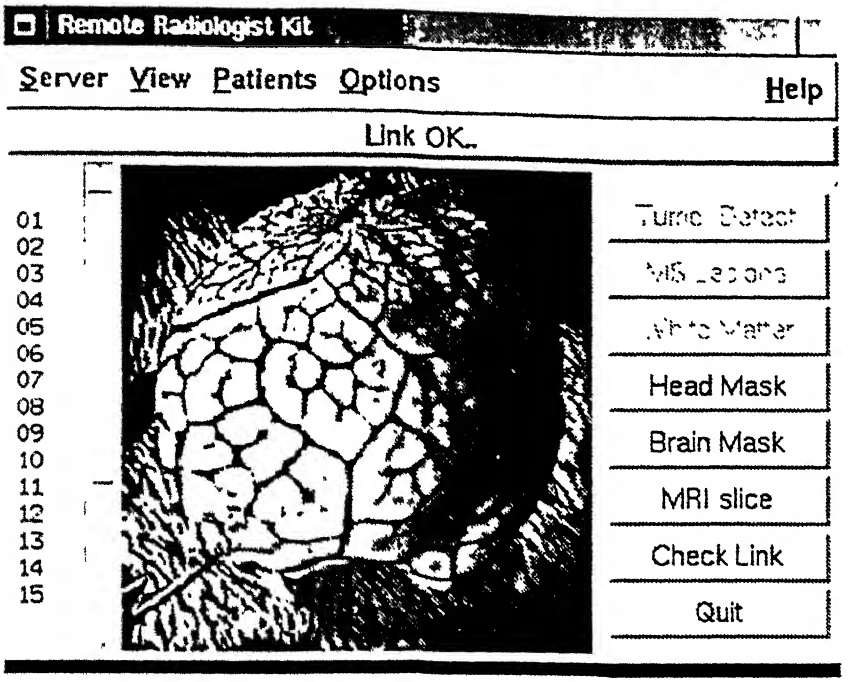


Figure 2.6: The ITP connection tested OK.

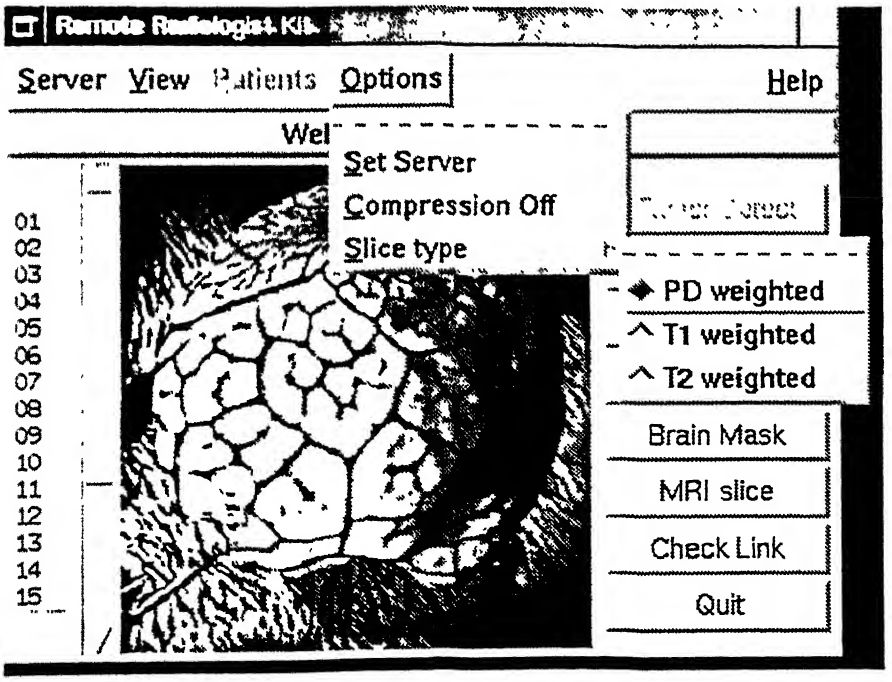


Figure 2.7: The `Slice type` menu option to change the type of slice (PD/T1/T2) to be viewed.

- `Head Mask` This button transfers and displays the head mask overlaid on the original slice according to current selections.

- `Brain Mask` This button transfers and displays the detected brain mask overlaid on the original slice according to current selections.

- Other `Server` menu commands: There are two additional remote server commands available on the `Server` menu for triggering segmentation computations by the server. The `Do Head` menu item (Figure 2.4) asks the server to carry out head mask detection computations. The `Do it all` menu item forces brain boundary detection for the whole MRI volume.

2.11 `View` options

There are a couple of viewing options provided through the pull-down `View` menu (Figure 2.8). The bounding box of a mask is shown if a masked slice (head mask or brain mask) is being displayed when the `Bounding Box` option is selected. Sometimes it is useful for the radiologist to look at the edges of the image to make an analysis. Therefore, an edge detection option is also provided in the `View` menu.

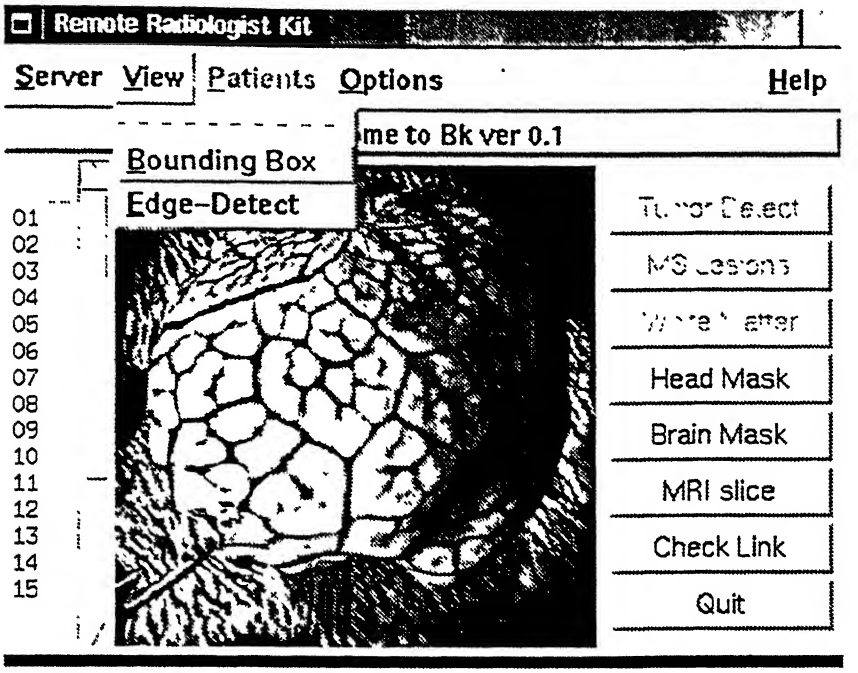


Figure 2.8: The View menu options.

2.12 Post-processing Buttons

On the right hand panel. three buttons corresponding to different post-processing algorithms are provided. As earlier mentioned in Chapter 1. since these have not been implemented in this thesis, these buttons are kept deactivated.

2.13 Image Compression

A knowledge-based compression scheme has been implemented in the ITP server. Since one of the prime purposes of carrying out intracranial boundary detection at the server end is to be able to do region-based compression, we have done so, in a very simple way.

If the image slice with the brain mask overlaid on it is to be transmitted, the ITP server determines the brain bounding box from the mask, and codes all non-brain pixels at half the greyscale depth compared to their original depth (e.g., if the pixels were originally represented by 8 bits, now they will be represented by 4 bits only). The brain region is transmitted at the original resolution.

Similarly, if the image slice with the head mask overlaid on it is to be transmitted, the ITP server determines the head bounding box and it codes the pixels outside the box at half the original resolution. This is done even while serving a request for original image slice.

This is a lossy compression scheme, but is capable of achieving high compression ratio without compromising image quality at the important portions of the image. The knowledge that the doctor is usually interested only in the brain region is utilized in a simple manner to produce compression. The results of such compression is discussed in more detail in Chapter 4.

Chapter 3

Brain Segmentation

3.1 Overview

An automatic method for segmenting the brain from head magnetic resonance (MR) images has been implemented on the server side of our teleradiology kit, Bk. This method uses an integrated approach which employs image processing techniques based on anisotropic filters, morphological operations, and *a priori knowledge*, which is used to remove the eyes and other non-brain regions. It is a multistage process, involving first removal of the background noise leaving a head mask, then finding a outline of the brain using the head mask information.

Recall from Chapter 1 that the primary goal of intracranial boundary detection in this thesis is to develop a preprocessing step that will allow for significant data compression for data transmission and improve the performance of MS lesion segmentation algorithms, brain tumor detection algorithms and other kinds of post processing algorithms normally used by the radiologist. These algorithms

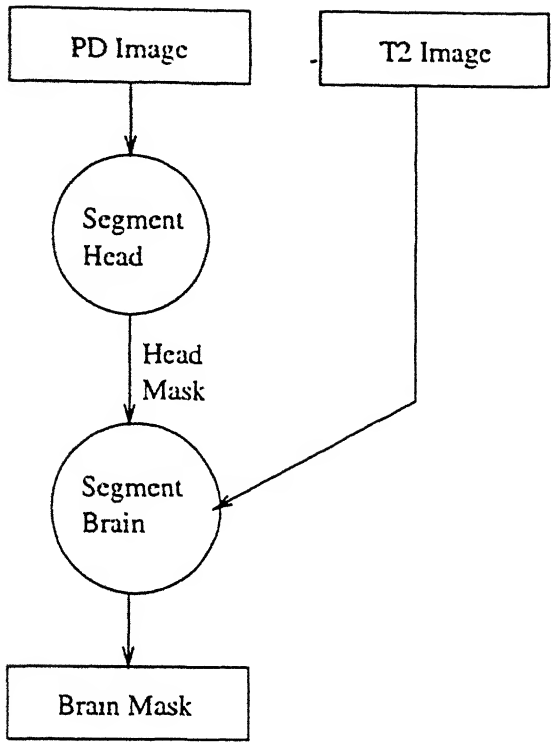


Figure 3.1: A simplified data flow diagram representing automatic intracranial boundary detection

have difficulty dealing with tissues outside the brain, such as skin and fat, because these tissues can have similar intensities as brain tissues and MS lesions. Consequently, the removal of tissues outside the intracranial boundary is necessary for successful segmentation.

We have implemented an automatic intracranial boundary detection method consisting of two main steps:

- Segment Head
- Generate Brain Mask using Head Mask information

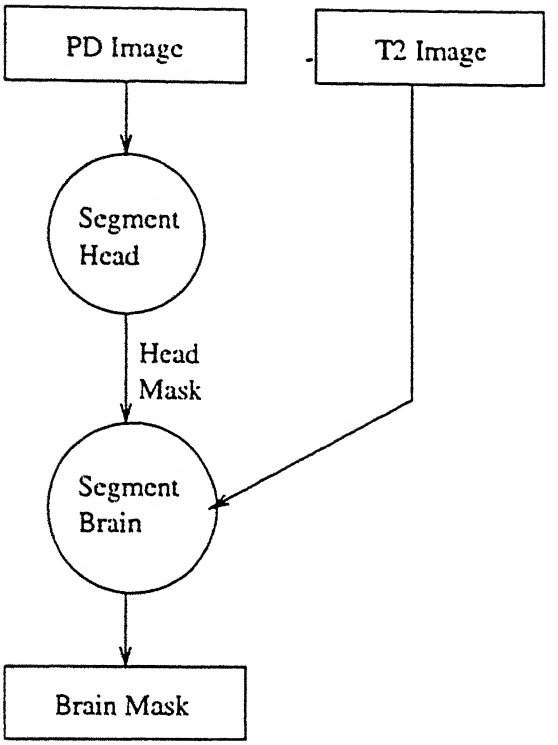


Figure 3.1: A simplified data flow diagram representing automatic intracranial boundary detection

have difficulty dealing with tissues outside the brain, such as skin and fat, because these tissues can have similar intensities as brain tissues and MS lesions. Consequently, the removal of tissues outside the intracranial boundary is necessary for successful segmentation.

We have implemented an automatic intracranial boundary detection method consisting of two main steps:

- Segment Head
- Generate Brain Mask using Head Mask information

The method assumes that PD-weighted and a T2-weighted MRI data sets have been simultaneously acquired.The steps are explained in detail in the following sections

3.2 Segment Head

The Segment Head process uses histogram analysis to automatically threshold data from background noise in MR images.

3.2.1 Histogram Image Volume

The PD-weighted MR image volume, which resides in a directory in the form of individual files representing individual slices is read in and a voxel intensity histogram is produced. Along with this, the standard deviation and mean of each intensity level is also calculated for the whole MR volume.

3.2.2 Determine Background Threshold Level

This step uses the method suggested by Brummer et al. to determine the "best" threshold level for removing background noise in PD-weighted MR images [12]. The background noise in PD-weighted MR scans has a Rayleigh distribution. The method fits a Rayleigh curve to the low intensity bins in the histogram using a Levenberg-Marquardt minimization algorithm [23]. The threshold is then chosen using error minimization. The subtraction of the best fit Rayleigh curve function,

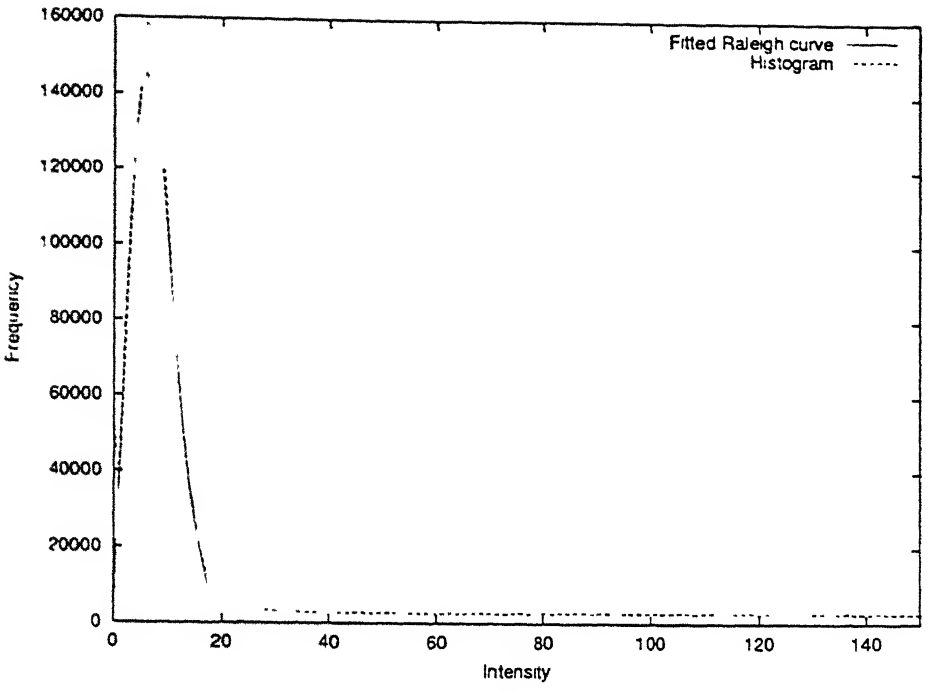


Figure 3.2: The best fit Rayleigh curve superimposed on the histogram of Data Set 2. In this case, $\tau = 20$. Note that the whole range of intensity levels (256) is not shown.

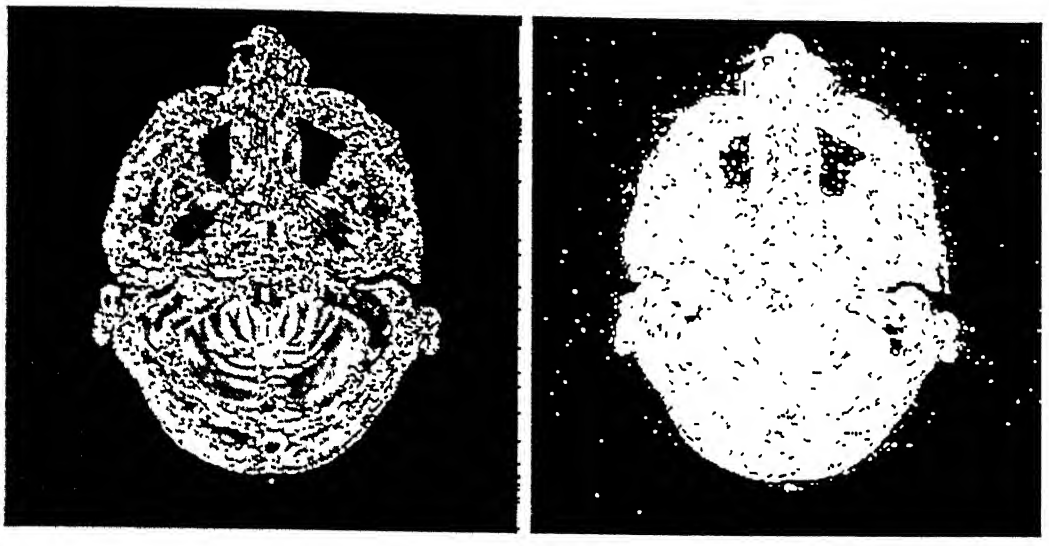
$r(f)$, from the volume histogram, $h(f)$, produces a bimodal distribution:

$$g(f) = h(f) - r(f) \tag{3.1}$$

A minimum error threshold, τ, can be determined from $g(f)$ by minimizing an error term, ϵ_τ:

$$\epsilon_\tau = \sum_{f=0}^{\tau-1} g(f) + \sum_{f=\tau}^{\infty} r(f) \tag{3.2}$$

Figure 3.2 shows the Rayleigh curve superimposed on the histogram of Data Set 2.

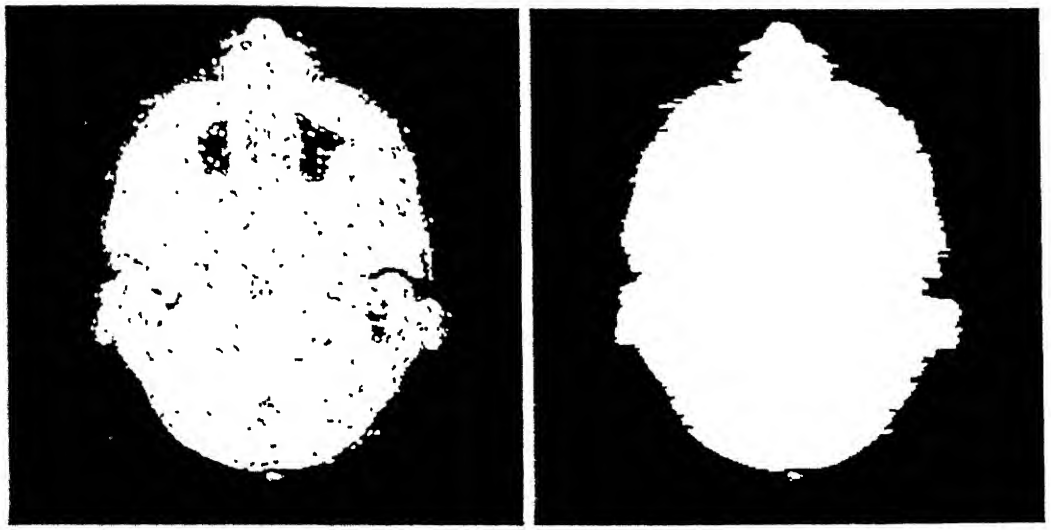


(a) (b)

Figure 3.3: An initial head mask produced using an automatic threshold. (a) Original image. (b) Initial head mask.

3.2.3 Binary Thresholding

Now, we threshold the grey-valued MR volume at τ, producing a binary image volume. The result of this threshold is shown in Figure 3.3 for slice 29 of Data Set 2. Notice that the binary image contains "speckle" outside the head region and that there are misclassified regions within the head. This "noise" is removed using morphological operations.



(a) (b)

Figure 3.4: Refining the head mask of Figure 3.3(b). (a) After Pepper-and-Salt Noise Removal. (b) Final Head Mask after row-scan algorithm.

3.2.4 Remove Noise

The sequence of morphological operations performed on the thresholded image is described below:

Pepper-and-Salt Noise Removal

Single-pixel objects in a background and single-pixel holes in an object of a binary image, referred to as pepper-and-salt noise, are often the result of propagation of uncorrelated noise in an original image through some binarization process. They can be detected as pixels with binary value different from all neighbors, and they can be removed by changing their binary value.

We extend this concept to two-pixel wide speckle by removing isolated two-

pixel wide objects in a background and holes in an object of a binary image. The result of this operation is shown in Figure 3.4(a).

Get the Head Mask

The next stage involves the detection of the head boundary. Operations like region filling, followed by binary erosion and subsequent binary dilation have been suggested by Atkins *et al.* [6]. However, these are expensive operations and since our algorithm should work in a networked environment, one of the goals has been to reduce computational complexity. Therefore, we use a simple row-scanning and filling algorithm to identify the head mask. Each row is scanned once. During the scan, the initial run of white pixels is filled by black pixels if the length of the run is less than 4. The next run of white is presumed to be the head mask and the algorithm goes on filling pixels until it encounters the last white pixel in the row, except that, as in the beginning of scan, a stray run of white less than 4 at the end of the row is filled with black pixels. The head boundary detected by this method is not accurate for some slices, but this doesn't matter much since we are going to use only the head mask bounding box information in later stages to identify the brain boundary. The result of this operation is shown in Figure 3.4(b).

3.3 Generate Brain Mask

To isolate the brain from the other head components, such as the eyes, skin, and fat, we need to highlight the brain tissue and then use a threshold to remove unwanted components. The Generate Brain Mask process uses a nonlinear

anisotropic diffusion filter to reduce the voxel intensities of tissues outside the brain. It then automatically thresholds the diffused volume and uses spatial information provided by the head mask to produce a mask of the brain. T2-weighted MR images are used since they exhibit good contrast between brain and non-brain tissues.

3.3.1 Nonlinear Diffusion Filtering

This is the first step in the "Segment Brain" stage. This step diffuses each slice of the image volume. The parameters of the filter were chosen such that pixel intensities in regions corresponding to brain tissue would be regularized while pixel intensities in other regions would be attenuated. The attenuation of non-brain voxels enables a single threshold to isolate the brain, even in the presence of RF inhomogeneity. The theory of diffusion filtering is briefly described in Section 3.4.

3.3.2 Automatic Threshold

This step automatically determines a threshold level for segmenting brain tissues from the diffused MR volume. It does so by fitting a gaussian function to the histogram of the volume, then choosing a threshold level at a specified number of standard deviations from the mean.

This method of automatic threshold detection was suggested by Brummer et al. [12]. They use the technique to segment the brain from non-brain tissues in unfiltered MR scans. They show that voxels inside the brain have normally

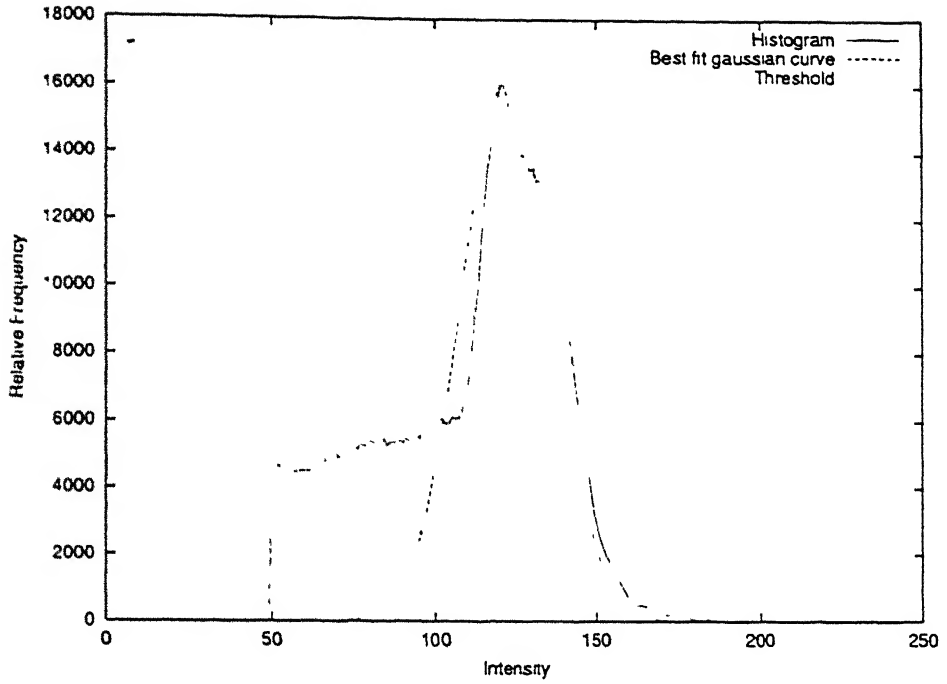
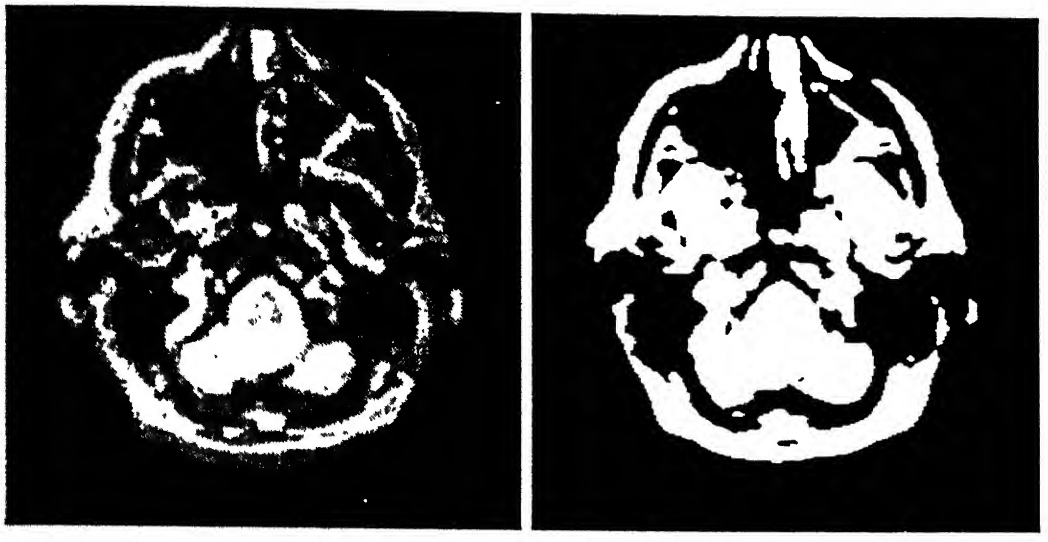


Figure 3.5: A histogram of the diffused T2-weighted MR scan with the best-fit Gaussian curve and threshold levels overlaid

distributed intensities, f , with standard deviation σ:

$$p_{brain}(f) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{f^2}{2\sigma^2}\right) \tag{3.3}$$

However, they have further shown that this is true only in case of PD-weighted MR scans. Brain voxel intensities in T2-weighted MR scans represent a somewhat normal, but skew intensity distribution. However, the regularization achieved by diffusion filtering results in a distribution that is close to normal. Figure 3.5 shows the voxel intensity histogram of the diffused Data Set 1 volume with the best fit gaussian curve overlaid. The threshold level is superimposed, which is taken 2 standard deviations below the mean for all cases. Using this threshold, each slice of the diffused MR volume is now binarized to produce a crude head mask. Figure 3.6 illustrates the results for slice 6 of Data Set 1.



(a) (b)

Figure 3.6: A binary mask produced by automatic thresholding. (a) The diffused image slice. (b) The corresponding binary mask.

3.3.3 Eliminate Misclassified and Non-Brain Regions

The binary mask includes non-brain regions. such as the eyes. and misclassified regions within the brain. These errors can be eliminated using morphology and spatial information. Non-brain regions are removed from the initial binary brain mask using a sequence of morphological operations described below:

Region filling

First, holes within each region in the binary mask are filled using a simple 10x10 square structuring element.

Erosion

Next, binary erosion is performed to separate the weakly connected regions. The mask used for erosion is a 10x10 structuring element with six off-pixels symmetrically placed at each of the four corners resulting in a hexagonal mask as shown below:

0	0	0	1	1	1	1	0	0	0
0	0	1	1	1	1	1	1	0	0
0	1	1	1	1	1	1	1	1	0
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1	1
0	1	1	1	1	1	1	1	1	0
0	0	1	1	1	1	1	1	0	0
0	0	0	1	1	1	1	0	0	0

Edge Detection and Thinning

Next, in the eroded image, the edge of each region is detected by using standard image gradient operation. The edges produced by this may be "thick". Therefore, thinning of edges are done so that edges of all the regions are one pixel thick.

Expert Elimination of Non-brain Regions

Once we have the thinned edges of all the regions, we detect each region by tracing its contour. Thus, we have the contours of all the regions. Next we remove non-brain regions using expert knowledge and spatial information from the head mask. We do so by eliminating features whose centroids fall outside the specified region within the head. Figure 3.7 illustrates the procedure. Rather than using a single brain bounding box for this purpose we use three different bounding boxes (with three different sets of parameters $minX, maxX, minY, maxY$) for three sets of slices in an MR volume. This is found to improve the performance in slices in which the brain tissue occupies a very small region. The value of parameters used were selected experimentally to produce consistent results for all slices of all data sets presented in this thesis.

Dilation

Once we have identified the region(s) corresponding to brain tissue, binary dilation of contours are performed with the same structuring element as used for

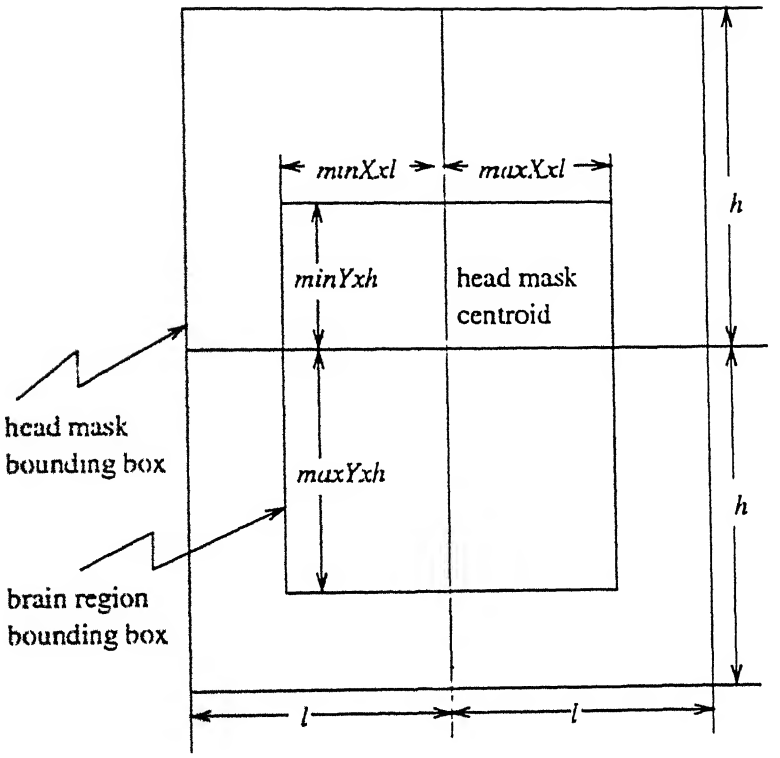


Figure 3.7: Spatial information from the head mask is used to eliminate regions that are unlikely to correspond to brain tissue. Closed contours whose centroids fall outside the *brain region bounding box* are discarded. In this case, $minX{=}0.3, maxX{=}0.75, minY{=}0.75, maxY{=}0.6$. These values are taken to be different for different set of slices depending on its location.

erosion shown earlier in this section to restore some of the peripheral tissues blackened by erosion. Now, the regions appear as thick rings whose outer boundaries are the ones which actually represent the intracranial boundary.

Get Brain Contour

Finally, edge detection and thinning are once again performed on the rings (regions) and the outer boundary of each ring thus detected represents the boundary of some brain tissue region in the MRI slice.

The whole process of expert elimination of non-brain regions from the brain mask using morphology and spatial information is shown in Figure 3.8 for slice 10 of Data Set 1.

3.4 Theory of Diffusion Filtering

Nonlinear anisotropic diffusion filters are iterative, tunable filters introduced by Perona and Malik [25]. Gerig et al. used such filters to enhance MR images [26]. Sapiro and Tannenbaum used a similar technique to perform edge preserving smoothing of MR images [27]. In the extreme case, such smoothing might produce a profile of RF inhomogeneity in the images. Others have shown that diffusion filters can be used to enhance and detect object edges within images [25].

Since these filters smooth or enhance MR images and detect edges, they might also be used for RF correction and/or intracranial boundary detection in MR

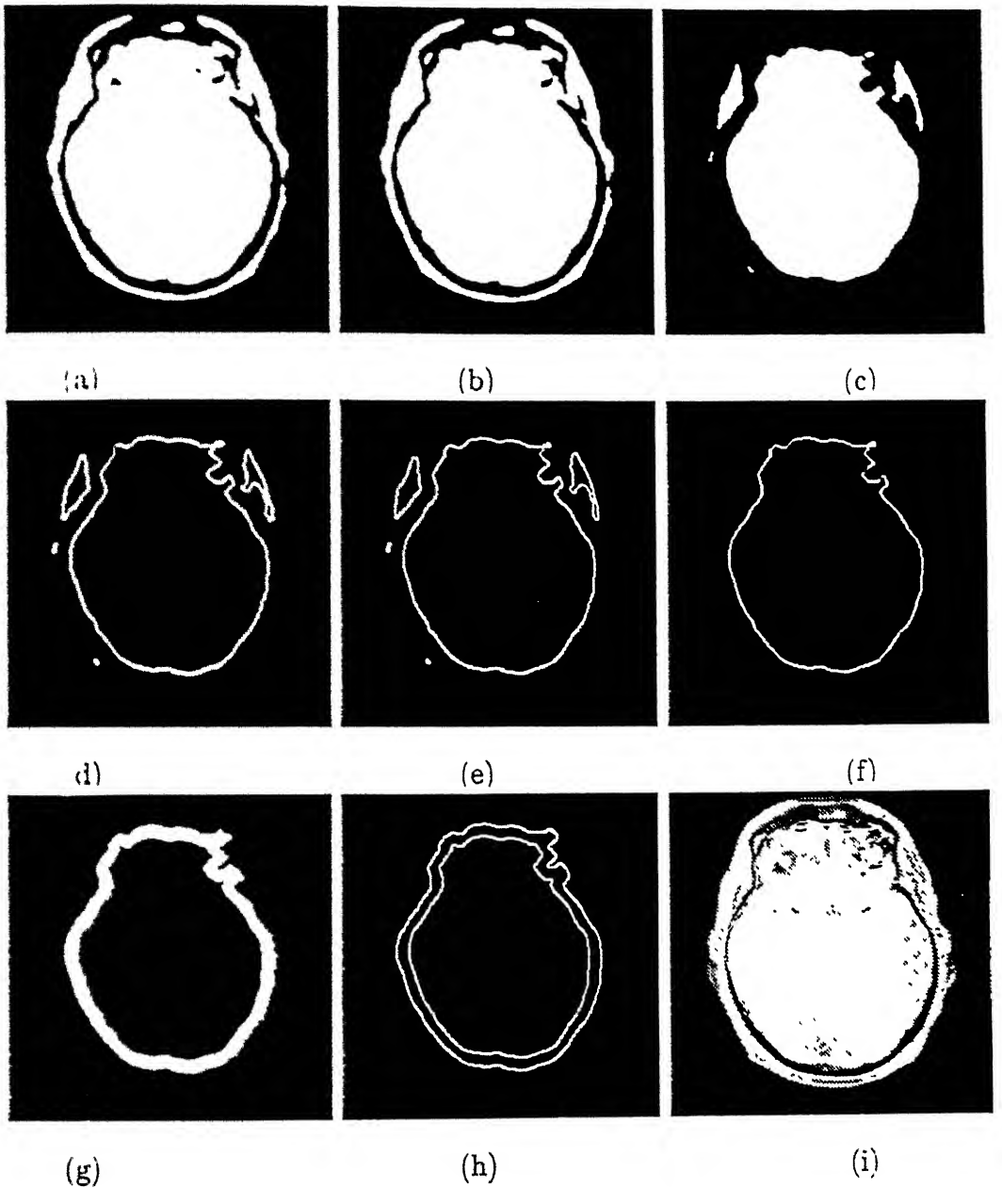


Figure 3.8: Morphology and spatial operations to segment brain. (a) Thresholded T2 image slice. (b) Regions filled. (c) After binary erosion. (d) After edge detection of all regions. (e) After edge thinning. (f) After expert elimination of non-brain regions. (g) After binary dilation. (h) After edge detection and thinning of dilated image. (i) Final brain boundary detected overlaid on the PD scan of the same slice.

images. To this end, nonlinear anisotropic diffusion filters are studied here.

3.4.1 Definition

Perona and Malik formulate the anisotropic diffusion filter as a diffusion process that encourages intra-region smoothing while inhibiting inter-region smoothing. Mathematically, the process is defined as follows:

$$\frac{\partial}{\partial t}(\bar{x}, t) = \nabla \bullet (c(\bar{x}, t)\nabla I(\bar{x}, t)) \tag{3.4}$$

In our case, $I(x; t)$ is the MR image. x refers to the image axes (i.e.$x; y; z$) and t refers to the iteration step. $c(\bar{x}; t)$ is called the *diffusion function* and is a monotonically decreasing function of the image gradient magnitude:

$$c(\bar{x}, t) = f(\mid \nabla I(\bar{x}, t) \mid) \tag{3.5}$$

It allows for locally adaptive diffusion strengths; edges are selectively smoothed or enhanced based on the evaluation of the diffusion function. We use the following diffusion function suggested in [25]:

$$c(\bar{x}, t) = \exp\left(-\left(\frac{\mid \nabla I(\bar{x}, t) \mid}{\kappa}\right)^2\right) \tag{3.6}$$

κ is referred to as the diffusion constant or the flow constant. Obviously, the behavior of the filter depends on κ. To clarify the effect of κ, and the diffusion function, on the diffusion process, it is helpful to define a flow function:

$$\Phi(\bar{x}, t) = c(\bar{x}, t) \cdot \nabla I(\bar{x}, t) \tag{3.7}$$

Equation 3.4 can be rewritten as:

$$\frac{\partial}{\partial t} I(\bar{x}, t) = \nabla \bullet (\Phi(\bar{x}, t)) \tag{3.8}$$

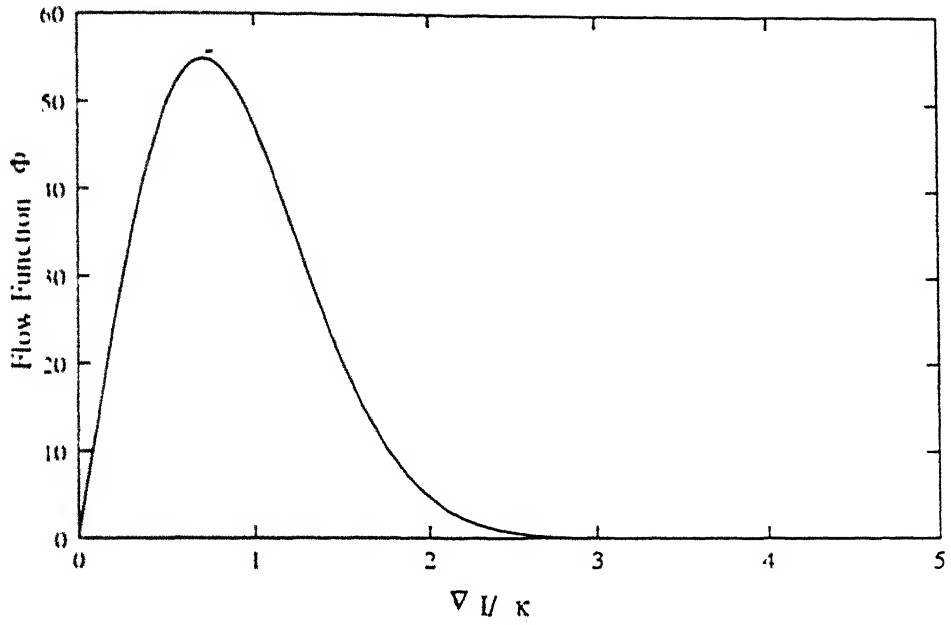


Figure 3.9: Flow function plotted as a function of image gradient

The flow function Φ corresponding to the diffusion function in Equation 3.6 is plotted in Figure 3.9.

Notice that flow increases with the gradient strength to the point where $\mid \nabla I \mid \simeq \kappa$, then decreases to zero. This behavior implies that the diffusion process maintains homogeneous regions (where $\mid \nabla I \mid \ll \kappa$) since little flow is generated. Similarly, edges are preserved because the flow is small in regions where $\mid \nabla I \mid \gg \kappa$.

The greatest flow is produced when the image gradient magnitude is close to the value of κ. Therefore, by choosing κ to correspond to gradient magnitudes produced by noise, the diffusion process can be used to reduce noise in images. Assuming an image contains no discontinuities, object edges can be enhanced by choosing a value of κ slightly less than the gradient magnitude of the edges.

3.4.2 Discrete Implementation

The discrete implementation of the nonlinear anisotropic diffusion filter is straightforward, the details of which are given in [24]. Here we shall write down the expression for the 2-D discrete implementation which we have used for filtering:

$$\begin{aligned}
\frac{\partial}{\partial t}I(x,y,t) &= \frac{\partial}{\partial x}\left[c(x,y,t)\cdot\frac{\partial}{\partial x}I(x,y,t)\right] \\
&\quad +\frac{\partial}{\partial y}\left[c(x,y,t)\cdot\frac{\partial}{\partial y}I(x,y,t)\right] \\
&\approx \frac{1}{(\Delta x)^2}\left[c\left(x+\frac{\Delta x}{2},y,t\right)\cdot(I(x+\Delta x,y,t)-I(x,y,t))\right. \\
&\quad \left.-c\left(x-\frac{\Delta x}{2},y,t\right)\cdot(I(x,y,t)-I(x-\Delta x,y,t))\right] \\
&\quad +\frac{1}{(\Delta y)^2}\left[c\left(x,y+\frac{\Delta y}{2},y,t\right)\cdot(I(x,y+\Delta y,t)-I(x,y,t))\right. \\
&\quad \left.-c\left(x,y-\frac{\Delta y}{2},t\right)\cdot(I(x,y,t)-I(x,y-\Delta y,t))\right] \\
&= \Phi_{east}+\Phi_{west}+\Phi_{north}+\Phi_{south}\,|_{\Delta x=\Delta y=1}
\end{aligned} \tag{3.9}$$

The filtering process consists of updating each pixel in the image by an amount equal to the flow contributed by its four nearest neighbors:

$$I(x,y,t+\Delta t)\approx I(x,y,t)+\Delta t\cdot(\Phi_{east}-\Phi_{west}+\Phi_{north}+\Phi_{south}) \tag{3.10}$$

where Δt, the iteration step size, is a filter parameter.

Chapter 4

Results and Discussion

4.1 Overview

In this chapter we shall present the results of applying our automatic intracranial boundary detection algorithm to the available data sets. We shall also illustrate a typical session with our teleradiology kit **Bk** and investigate its performance.

4.2 Data sets

Data set acquisition has been a major problem right from the beginning of our work. First, the Visual Human Data [2] were acquired but were later on found to be unsuitable. Then, we could finally acquire 2 data sets from the Simon Fraser University, Burnaby, Canada. Data Set 1 has 22 slices, with both PD and T2-weighted scans available. Data Set 3 has 20 slices, also with both PD and

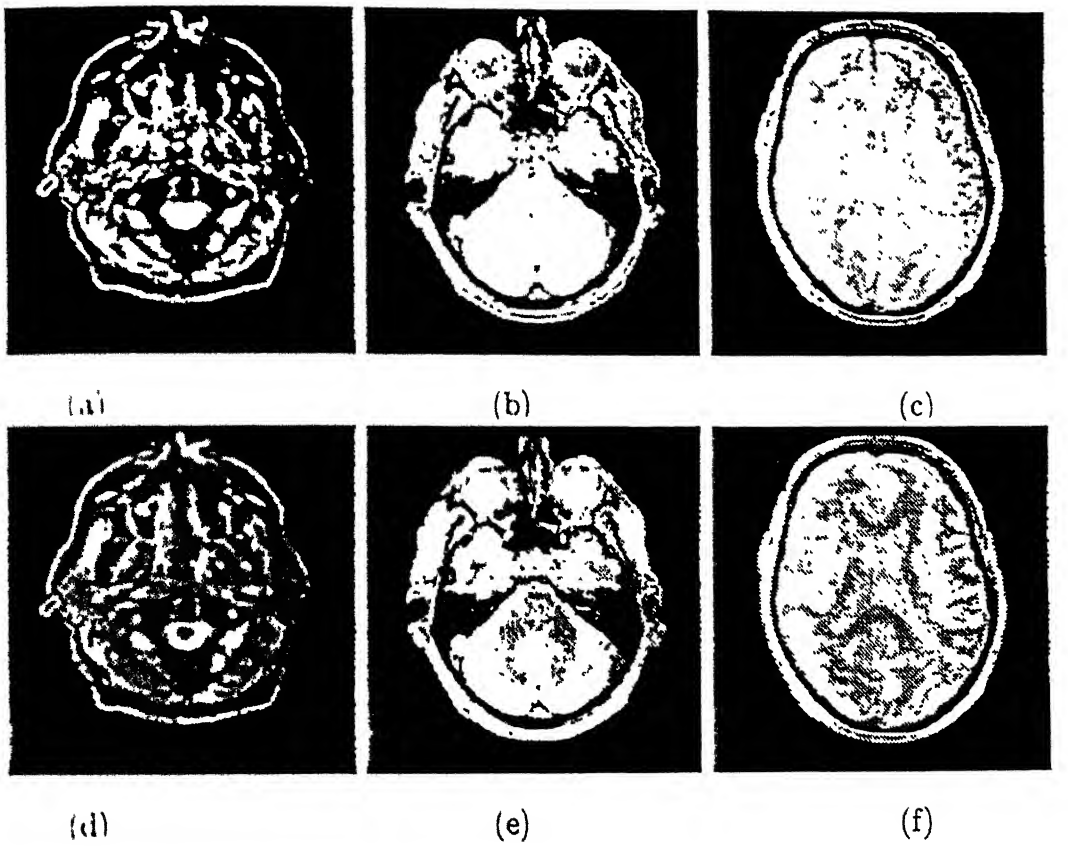


Figure 4.1: Selected Slices from MRI Data Set 1. (Top) PD-weighted. (Bottom) T2-weighted. (a),(d)Slice 1. (b),(e)Slice 8. (c),(f) Slice 15.

T2-weighted scans available. Data Set 2 is the Visible Human Data in which 33 slices of all the PD,T1 and T2-weighted scans are available. But this was not used for brain segmentation. Some results of head segmentation on this Data Set are shown in Chapter 3.

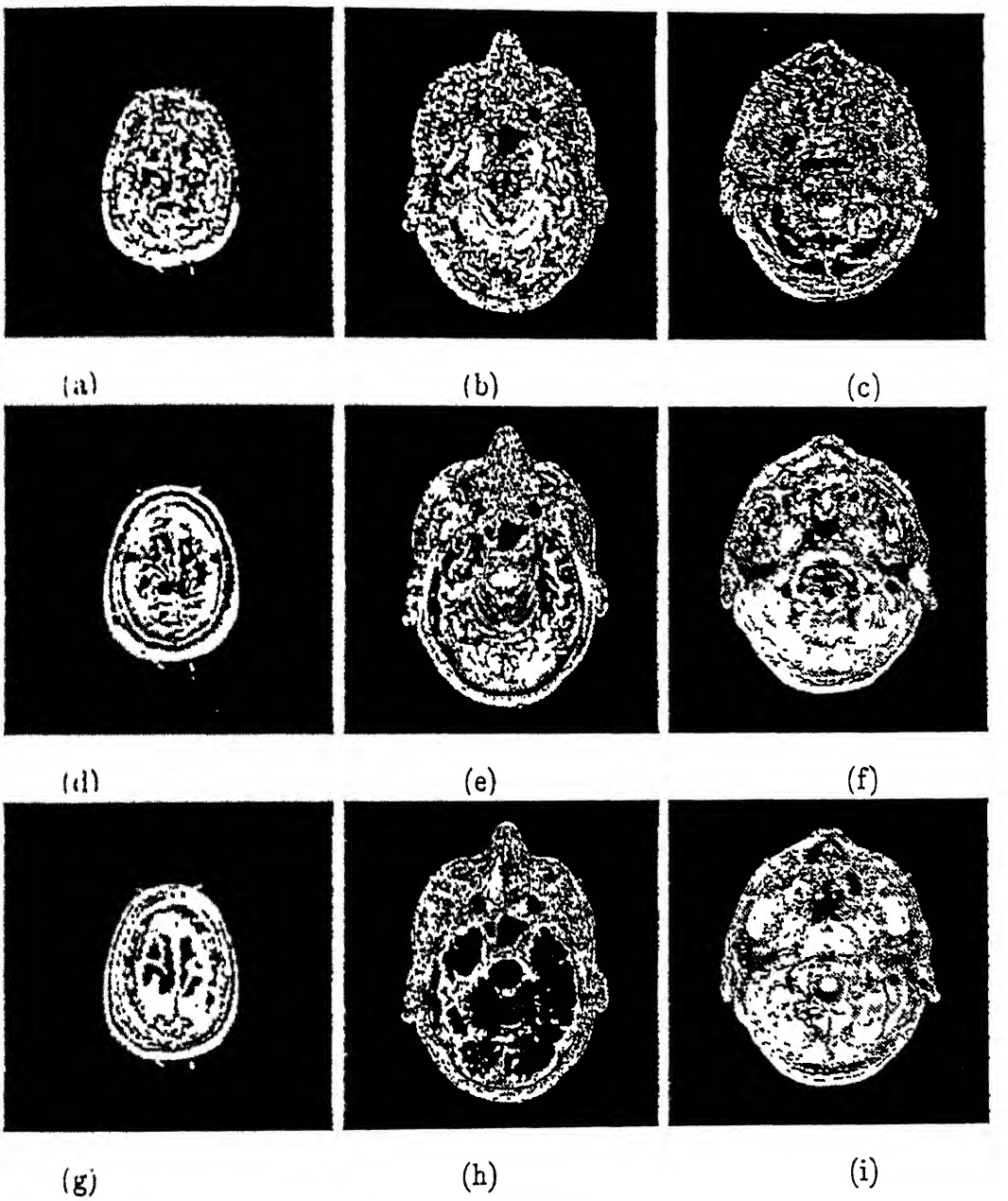


Figure 4.2: Selected Slices from MRI Data Set 2. (Top) PD-weighted. (Middle) T2-weighted. (Bottom) T1-weighted. (a),(d),(g)Slice 1. (b),(e),(h)Slice 8. (c),(f),(i)Slice 15.

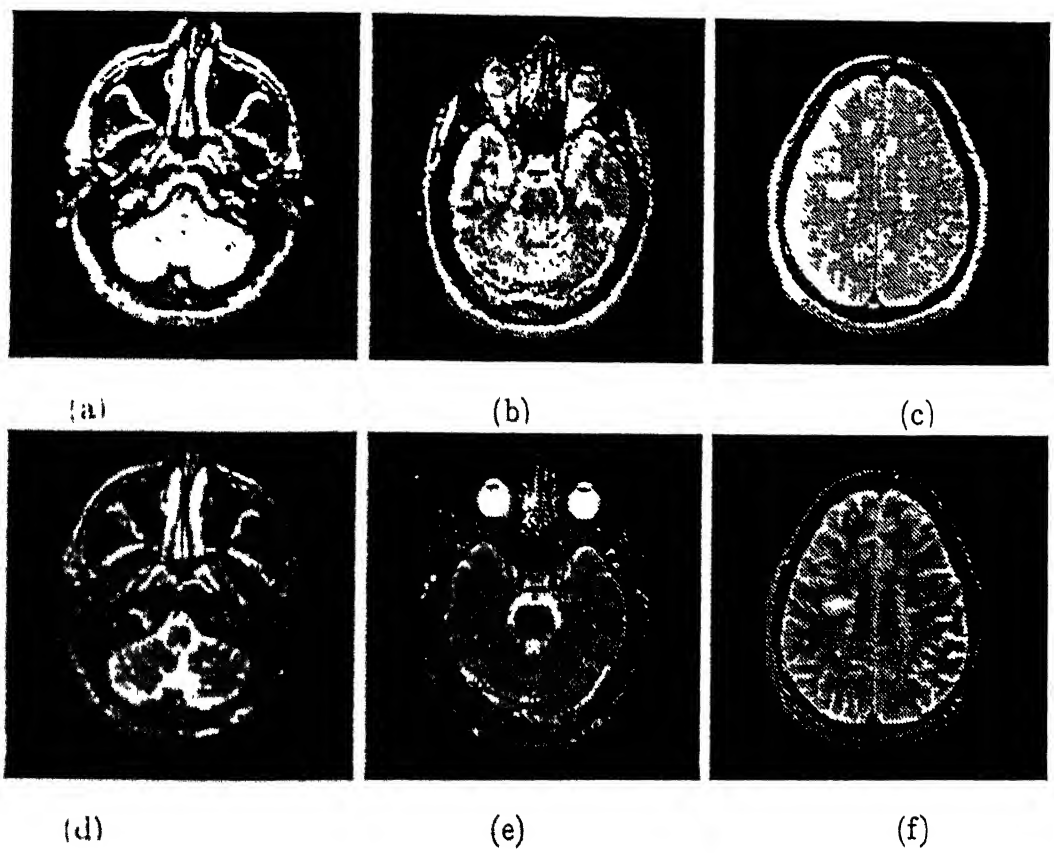


Figure 4.3: Selected Slices from MRI Data Set 3. (Top) PD-weighted. (Bottom) T2-weighted. (a),(d)Slice 4. (b),(e)Slice 9. (c),(f) Slice 19.

4.3 Results of Head Segmentation

In order to detect the intracranial boundary, the first step is to identify a head mask from the PD-weighted MR volume. The sequence of steps for doing so have been detailed in Section 3.2. Results of head mask detection for two data sets are shown in Figures 4.4 and 4.5.

Note that the head boundary detected in some cases appear to be a discontinuous contour. This is so because the boundary points are identified in a single scan through the image to reduce computational complexity. Since each row is scanned exactly once, there are two boundary points in each row, because of which, *kinks* are not traced properly. However, this doesn't defeat the purpose of head mask detection, since all we want from the head mask is the head bounding box which is used as a priori knowledge for brain contour detection. The head mask will still, almost always, be the same even if the head boundary detected is discontinuous. In fact, in doing so, we don't lose any information as explained above, rather gain in terms of faster computation, which is crucial in a networked environment.

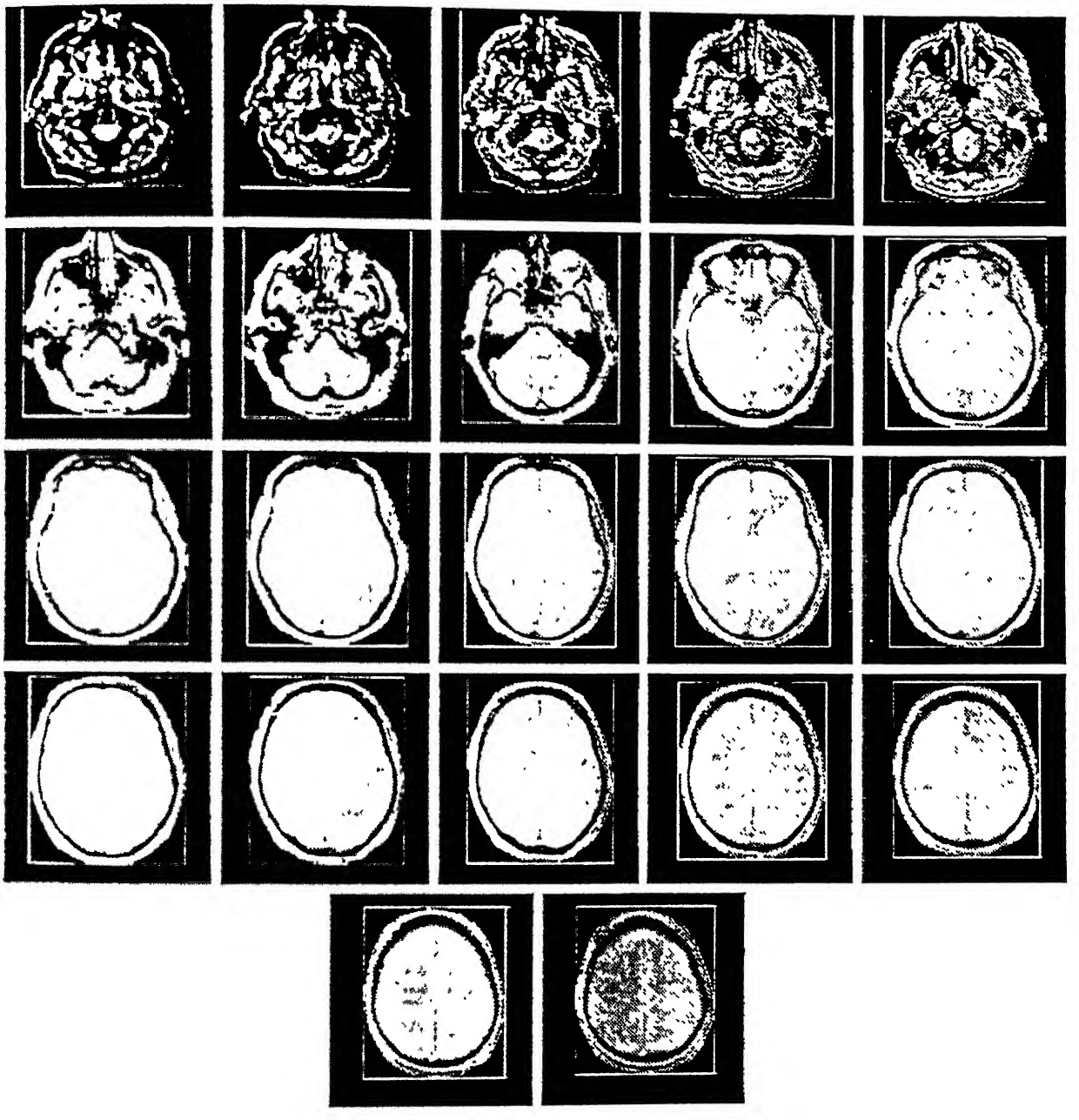


Figure 4.4: The Head Mask as well as the Head Bounding Box for MRI Data Set 1 overlaid on the PD-weighted scan

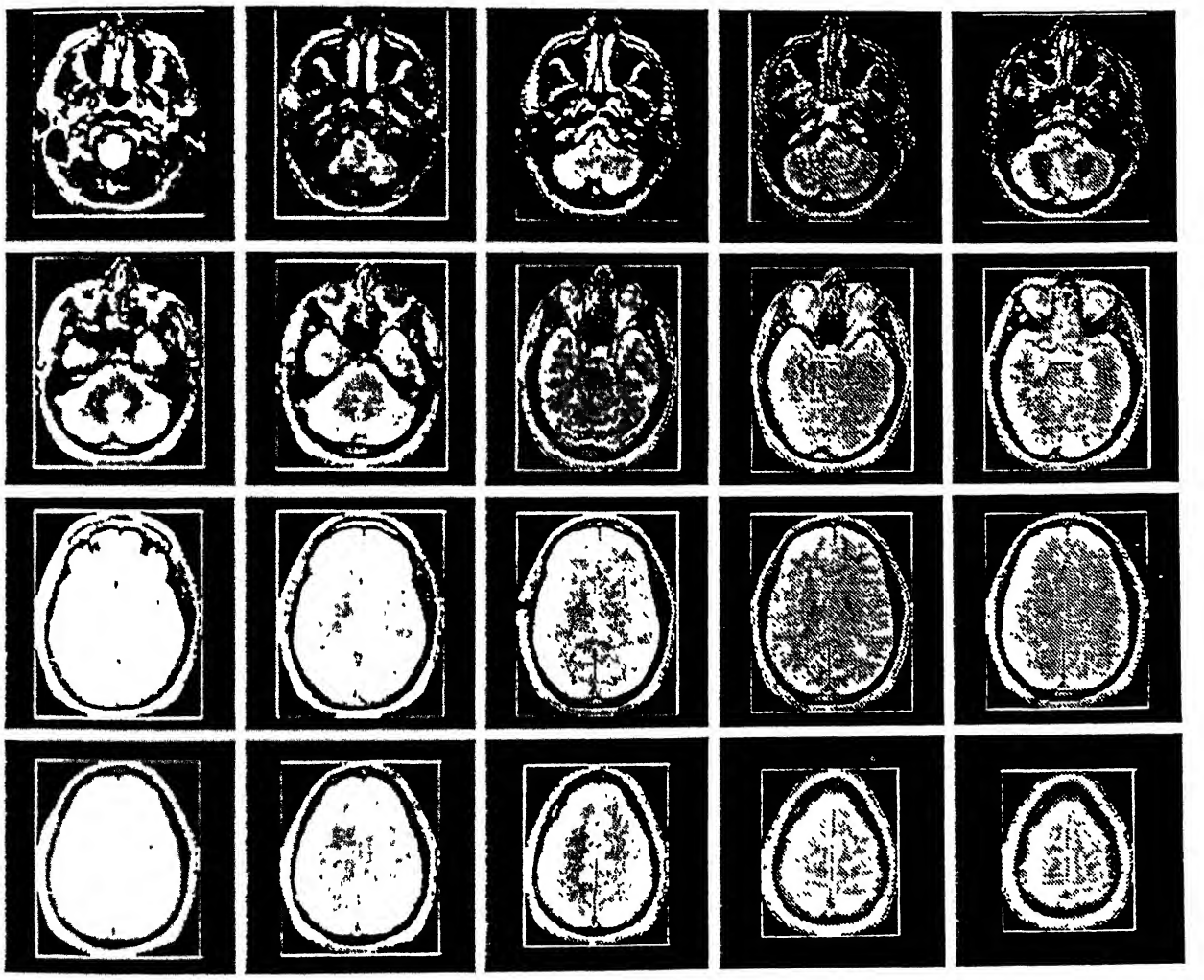


Figure 4.5: The Head Mask as well as the Head Bounding Box for MRI Data Set 3 overlaid on the PD-weighted scan

4.4 Results of Intracranial Boundary Detection

Once we have the head mask information for each slice in an MR volume, we apply non-linear anisotropic diffusion filtering on the T2-weighted volume (See Section 3.1). The filter parameters used in all the cases are: $\kappa = 128.0$ and $\Delta t = 0.19$. 20 iterations of filtering were carried out in each case. These numbers were all experimentally determined to give best results with all the data available.

After this, the filtered data is binarized by automatic thresholding and a sequence of morphological operations are applied to the resulting binary image to extract the brain contour. These methods have been discussed in detail in Section 3.3. Here we present results of brain boundary detection for Data Sets 1 and 3. (Figures 4.6, 4.7, 4.8 and 4.9).

4.5 Discussion on Brain Segmentation

As we can see from the figures, the boundary detection algorithm performs very well in almost all the slices. The few cases of misclassification are: In slice 22

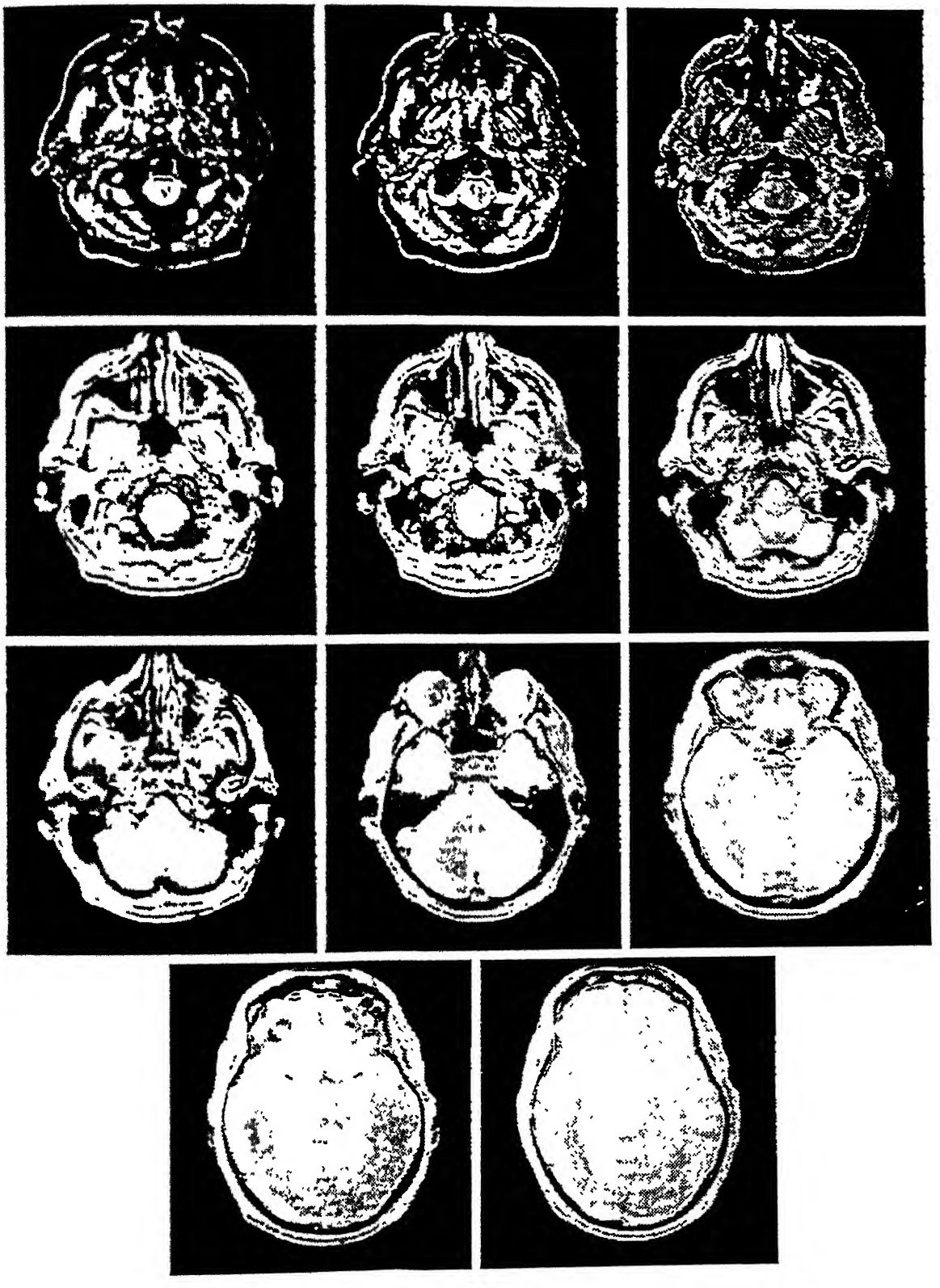


Figure 4.6: The Brain Mask for MRI Data Set 1 overlaid on the PD-weighted scan: Slices 1-11.

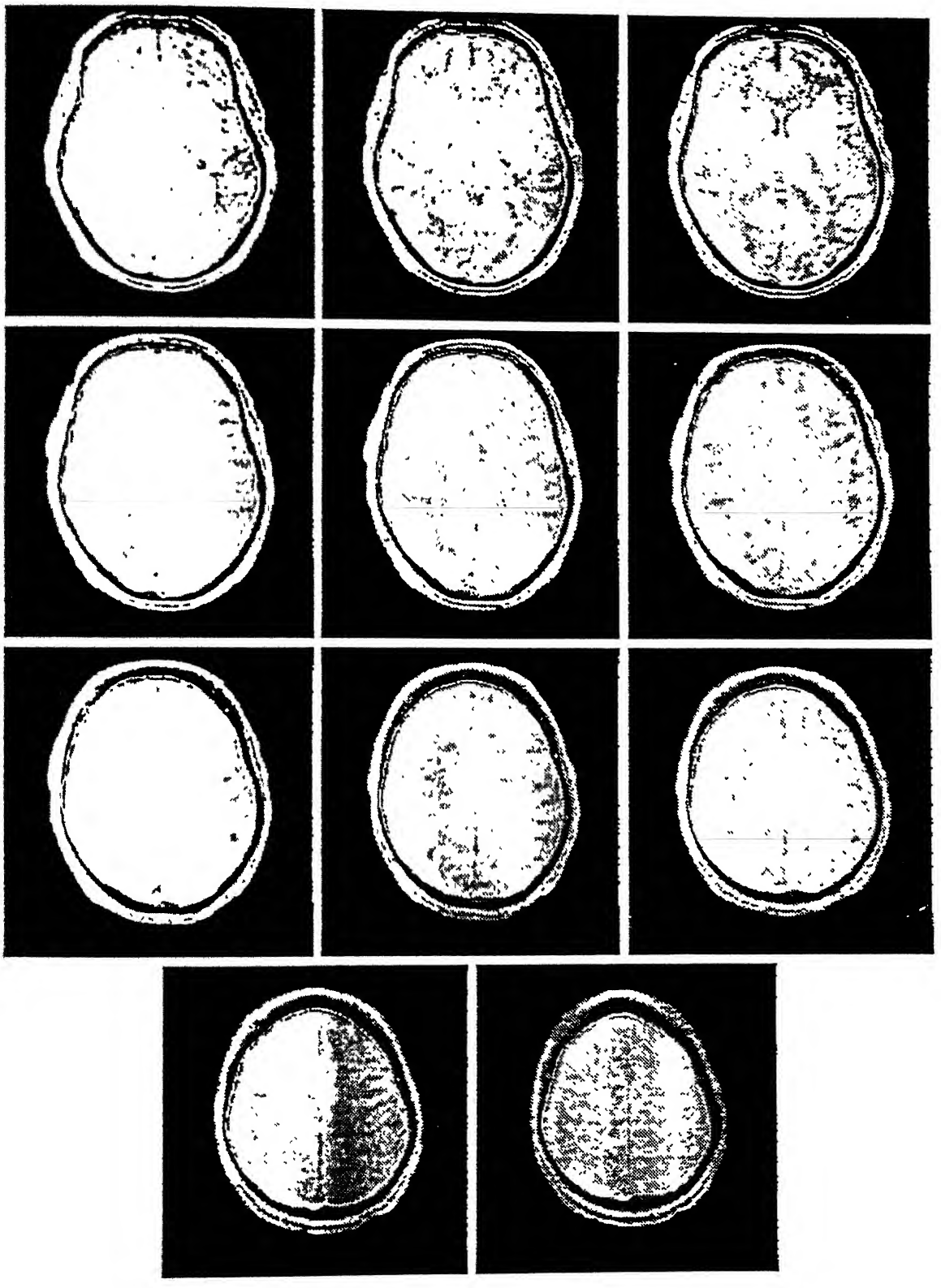


Figure 4.7: The Brain Mask for MRI Data Set 1 overlaid on the PD-weighted scan: Slices 12-22.

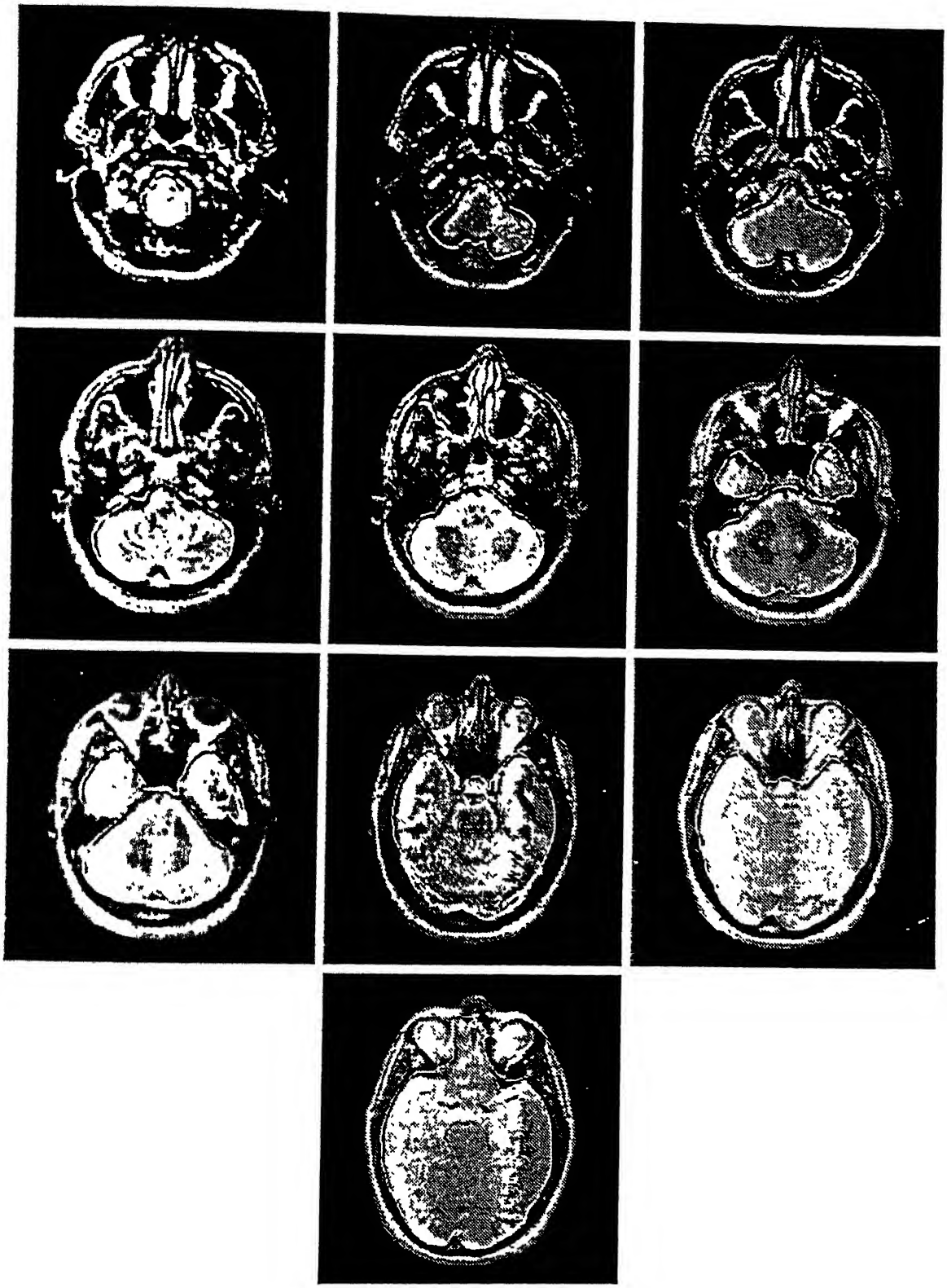


Figure 4.8: The Brain Mask for MRI Data Set 3 overlaid on the PD-weighted scan: Slices 1-10.

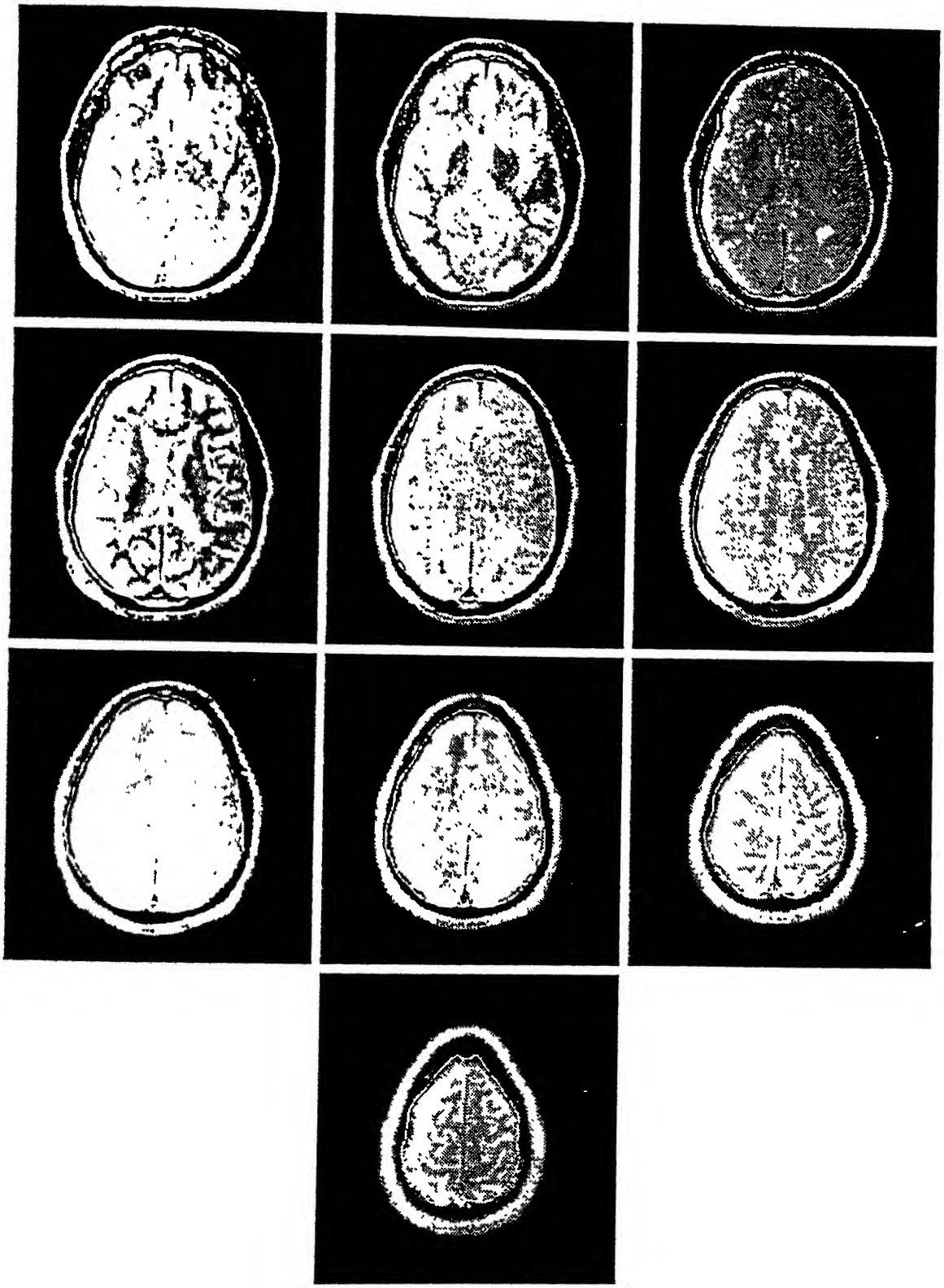


Figure 4.9: The Brain Mask for MRI Data Set 3 overlaid on the PD-weighted scan: Slices 11-20.

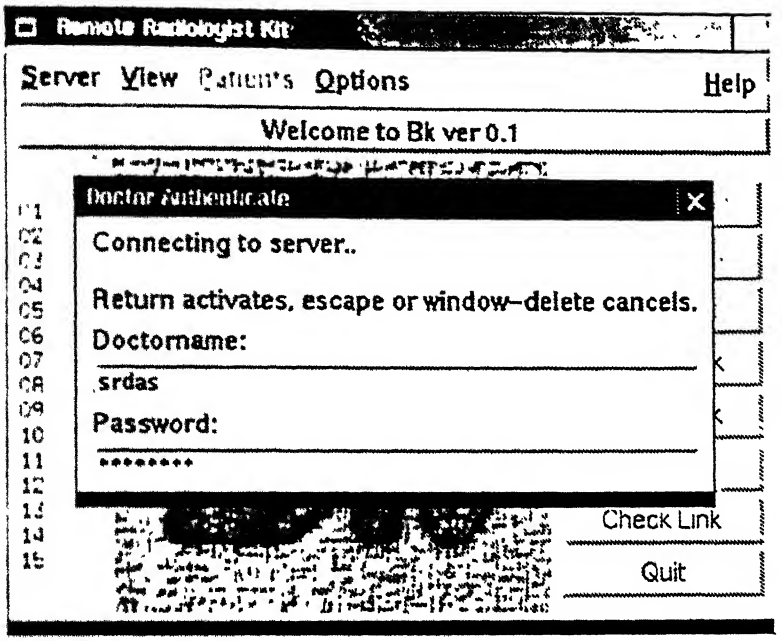


Figure 4.10: Authentication dialogue box at startup

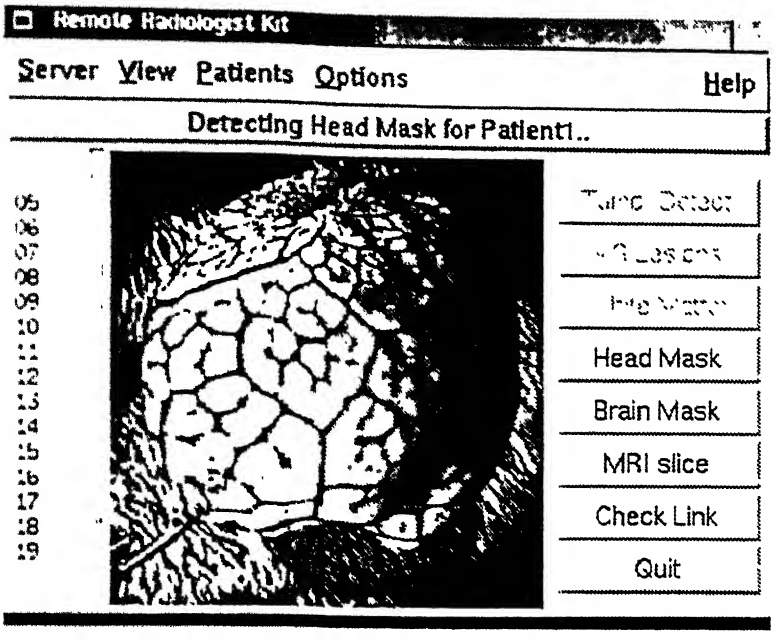


Figure 4.11: The Server->Do Head command has been invoked after selecting slice 17 from the listbox and Patient1 from the Patients menu.

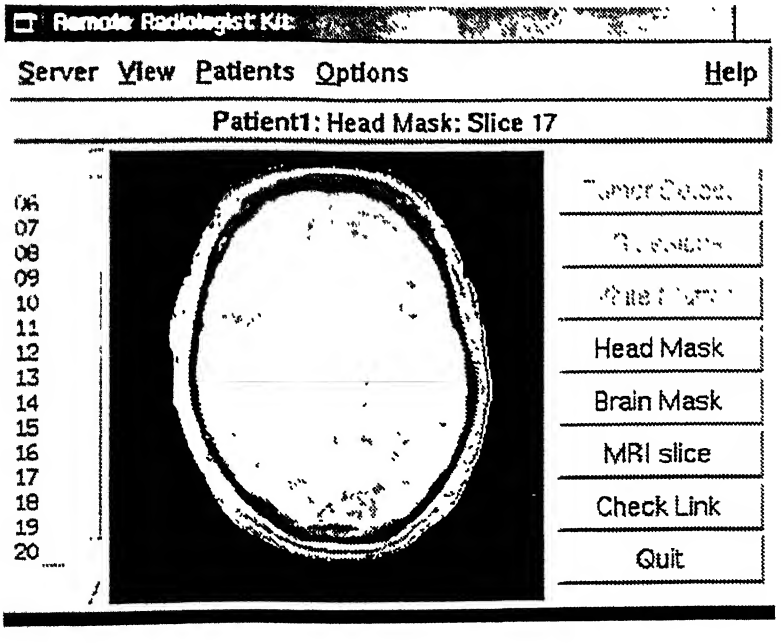


Figure 4.12: After Head Mask detection, the 17th slice for Patient1 has been transferred and displayed on the display area.

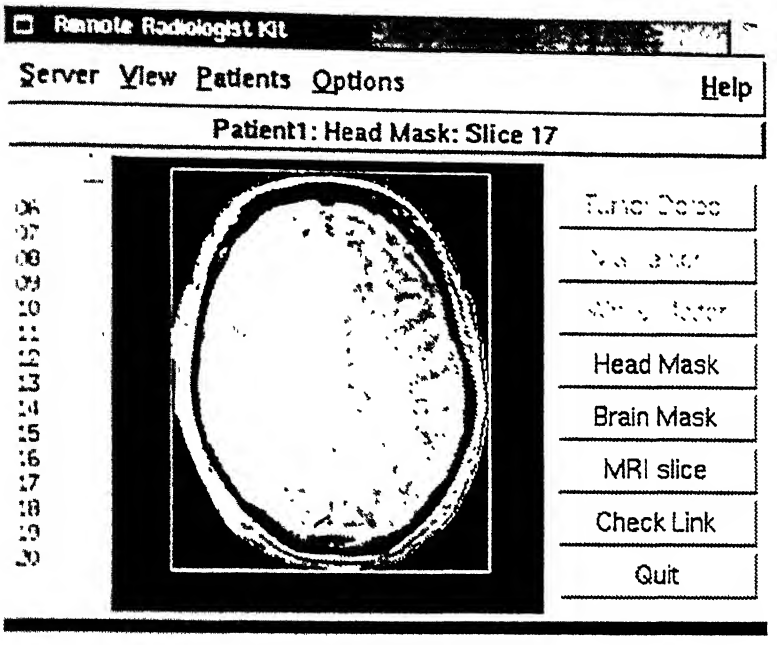


Figure 4.13: The doctor has selected the `View->Bounding Box` option so that the head bounding box is being displayed.

4.7 Discussion on Teleradiology Kit

The kit was tested on the Local Area Network(LAN) in the IIT campus, and the test sessions ran without any problem over our ethernet-based TCP/IP network.

In Figure 4.16, the 7th slice of Data Set 1 with detected brain contour overlaid is displayed on Bk after transmission without knowledge-based compression. In Figure 4.14, the compression scheme suggested in Section 2.13 is employed. The amount of data compression thus achieved is 41.83%. This is a huge compression, whereas there is no appreciable difference in image quality in the two figures. In fact, since the non-brain portion is seldom important for radiologist, one can go for very high compression ratios based on the bounding-box region-based compression. This is what has been done in Figure 4.15, where the region outside the brain bounding box has been coded by only 2 bits. In this case, the compression

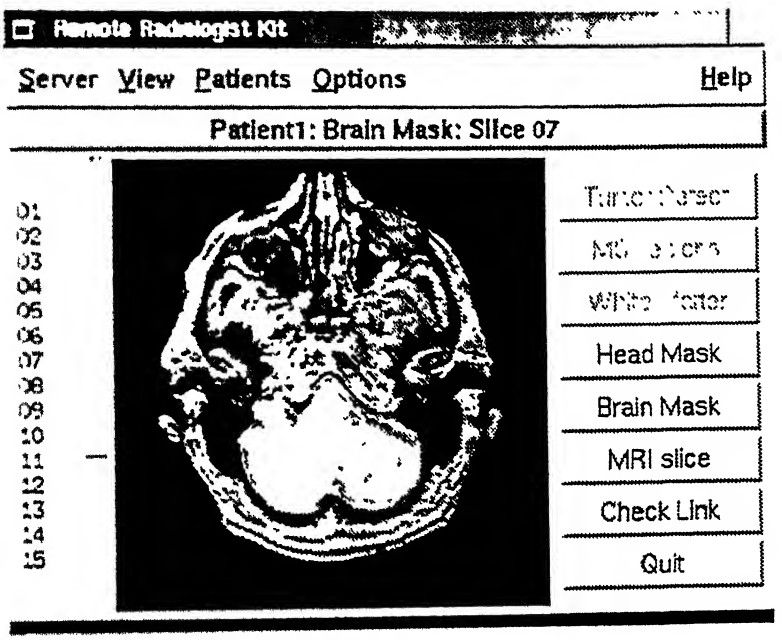


Figure 4.14: After selecting the 7th slice from the listbox, `Brain Mask` button has been pressed. The command for doing head contour detection is automatically issued to the server and the selected slice with brain mask overlaid is transferred and displayed.

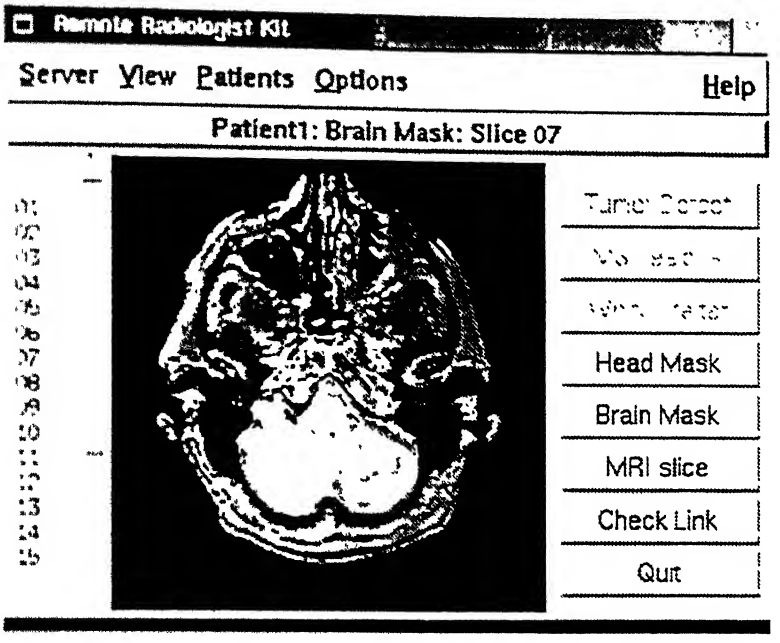


Figure 4.15: The 7th slice with brain mask overlaid, same as Figure 4.14, but with a very high compression ratio.

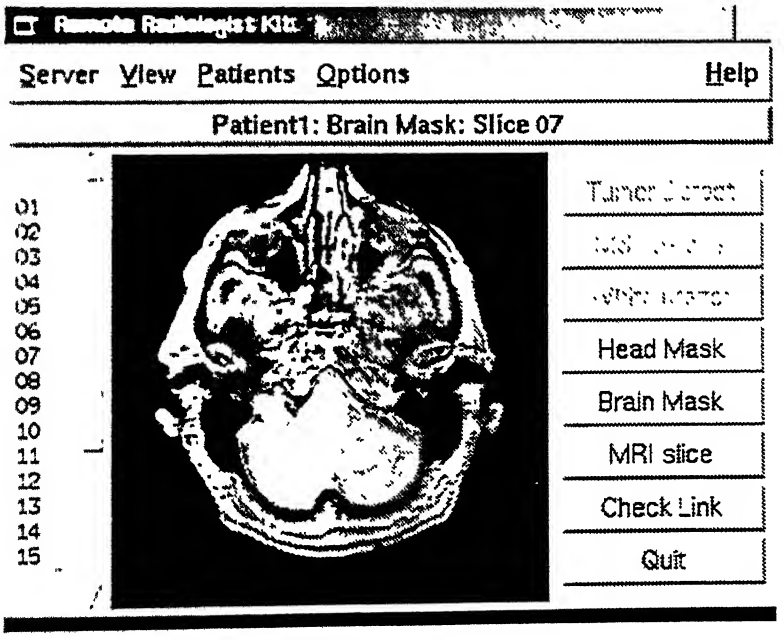


Figure 4.16: The 7th slice with brain mask overlaid, same as Figure 4.14, but with compression turned off from the `Options->Compression off` menu option.

ratio is as high as 62.74%. The amount of compression will of course, vary, as the brain bounding box changes in size from one slice to another.

4.8 Scope for Future Work

There are numerous things that could not be done in this thesis and could be valuable additions to **Bk**. both in terms of better GUI and server design and the brain segmentation algorithm improvement. Some suggestions are given below:

1. The post-processing algorithms for which buttons have been designed but kept deactivated may be developed and incorporated.
2. A report generation utility along with printing option for the radiologist can be incorporated.
3. 3-D visualization of MRI volume could be implemented using the popular extension of Tk called Vtk (Visual Toolkit).
4. **Bk** should be tested on Wide Area Networks (WAN) and on the internet.
5. The algorithm for brain segmentation should be made more robust so that the rate of misclassification is reduced. Also, it should work properly with high resolution scans.
6. The algorithm should be tested with more data and the results should be validated by a human radiologist.

Bibliography

[1] S. Laxminarayan, M. Majidi, D. Yatcilla, M. Jessup and R. Fekete, "The benefits of NII: The digital nirvana for health and medical research," in *Proc. Int. Workshop Mechatronics Surgery*, P. Brett, Ed. Bristol, U.K.: Univ. Bristol, 1995.

[2] Visible Human Data, *http://www.nlm.nih.gov/research/visible*

[3] Visible Embryo Project, *http://www.muritech.com/visembryo*

[4] Uwe Engelmann *et al.*, "A Three-Generation Model for Teleradiology," *IEEE Tran. on Information Technology in Biomedicine*, Vol. 2, No. 1, pp. 20-24, March 1998.

[5] Pinkesh J. Shah, Ralph Martinez and Bernard P. Zeigler, "Design, Analysis, and Implementation of a Telemedicine Remote Consultation and Diagnosis Session Playback Using Discrete Event System Specification," *IEEE Tran. on Information Technology in Biomedicine*, Vol. 1, No. 3, pp. 179-188, Sep. 1997.

[6] M. Stella Atkins and Blair T. Mackiewich, "Fully Automatic Segmentation of the Brain in MRI," *IEEE Trans. on Medical Imaging*, Vol. 17, No. 1, pp. 98-107, Feb. 1998.

[7] B. Johnston, M.S. Atkins and K.S. Booth, "Partial volume segmentation in 3-D of lesions and tissues in magnetic resonance images," *Proc. SPIE-Medical Imaging 1994*, Bellingham, WA, 1994, vol. 2167, pp. 28-39.

[8] Pamela A. Cosman, Robert M. Gray, and Richard A. Olshen, "Evaluating quality of compressed medical images: SNR, subjective rating, and diagnostic accuracy," *Proceedings of the IEEE*, 82(6):919-932, June 1994.

[9] Eve A. Riskin, Tom Lookabaugh, Philip A. Chou, and Robert M. Gray, "Variable rate vector quantization for medical image compression," *IEEE Transactions on Medical Imaging*, 9(3):290-298, September 1990.

[10] L.P. Clarke, R.P. Velthuizen, M.A. Camacho, J.J. Heine, M. Vaidyanathan, L.O. Hall. R.W. Thatcher, and M.L. Sibliger, "MRI segmentation: Methods and applications," *Magn. Reson. Imag.*, Vol. 13, No. 3. pp. 343-368, 1995.

[11] H. Suzuki and J. Toriwaki, "Automatic segmentation of head MRI images by knowledge guided thresholding," *Computerized Med. Imag. Graphics*, Vol. 15, No. 4, pp. 233-240. July-Aug, 1991.

[12] M.E. Brummer, R.M. Mersereau, R.L. Eisner, and R.R.J. Lewine, "Automatic detection of brain contours in MRI data sets", *IEEE Trans. Med. Imag.*, Vol 12, pp. 153-166, June 1993.

[13] A. Chakraborty, L.H. Staib, and J.S. Duncan, "An integrated approach to boundary finding in medical images," in *Proc. IEEE Workshop on Biomedical Image Analysis*, Los Alamos, CA, June 1994, pp. 13-22.

[14] A. Lundervold and G. Storvik, "Segmentation of brain parenchyma and cerebrospinal fluid in multispectral magnetic resonance images," *IEEE Trans. Med. Imag.*, Vol. 14, pp. 339-349, June 1995.

[15] T. Kapur, E.L. Grimson, W.M. Wells III, and R. Kikinis, "Segmentation of brain tissue from magnetic resonance images," *Med. Imag. Anal.*, Vol. 1, No. 2, 1996.

[16] W.M. Wells III, W.E.L. Grimson, R. Kikinis, and F.A. Jolesz, "Adaptive segmentation of MRI data," *IEEE Trans. Med. Imag.*, Vol. 15, pp. 429-443, Aug. 1996.

[17] H.E. Cline, W.E. Lorensen, R. Kikinis, and F. Jolesz, "Three-dimensional segmentation of MR images of the head using probability and connectivity," *J. Comput. Assist. Tomogr.*, Vol. 14, No.6, pp. 1037-1045, Nov./Dec., 1990.

[18] F. Pannizz *et al.*, "Quantitative MRI studies for assessment of multiple sclerosis," *Magn. Reson. Med.*, Vol. 24, pp. 90-99, 1992.

[19] Sun Microsystems, "Network Programming Manual Release 4.0"

[20] Uresh Vahalia, "UNIX Internals : The New Frontiers," Prentice Hall, New Jersey.

[21] S. Leffler, W. Joy, R. Fabry, M. Karels, "Networking Implementation Notes-4.3 BSD Edition," *University of california, Berkeley*, CA, Apr. 1986.

[22] John K. Ousterhout, "Tcl and the Tk Toolkit," Addison-Wesley Publishing Company, ISBN 0-201-63337-X.

[23] W.H. Press, S.A. Teulolsky, W.T. Vetterling, and B.P. Flannery, "Numerical Recipes in C, The Art of Scientific Computing," Cambridge University Press, Cambridge, MA, 2nd edition, 1992.

[24] B. Mackiewich, "Intracranial boundary detection and radio frequency correction in magnetic resonance images," M.S. thesis, Simon Fraser Univ., Burnaby, B.C., Canada, Aug. 1995.

[25] Pietro Perona and Jitendra Malik, "Scale-space and edge detection using anisotropic diffusion," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 12(7):629-639, July 1990.

[26] Guido Gerig, Olaf Kubler, Ron Kikinis, and Ferenc A. Jolesz, "Nonlinear anisotropic filtering of MRI data," *IEEE Transactions on Medical Imaging*, 11(2):221-232, June 1992.

[27] Guillermo Sapiro and Allen Tannenbaum, "Edge preserving geometric smoothing of MRI data," Technical report, University of Minnisota, April 1994. Dept. of Electrical Engineering.

[28] Mikael J. Simovits "The Des : An Extensive Documentation and Evaluation of the Data Encryption Standard (Cryptographic Series , No 68)," *http://www.amazon.com.*